**MASTER EN SCIENCES INFORMATIQUES**

# Étude concernant le remplacement d'un système d'information auprès du Conseil des Ministres CEE

Saitta, Carmelo

*Award date:*
1991

*Awarding institution:*
Universite de Namur

Facultés Universitaires
Notre Dame de la Paix-Namur

Institut d'informatique

Année académique 1990-1991

# Etude concernant le remplacement d'un système d'information auprès du Conseil des Ministres CEE.

Carmelo SAITTA

Promoteur: Professeur J. Fichefet

Mémoire présenté en vue
de l'obtention du titre de
*Licencié et Maître en Informatique*

## RESUME

L'actuel système d'information (S.I.) du Conseil des Ministres des Communautés Européennes arrive en fin d'exploitation. Pour cette raison, les responsables du service informatique se penchent sur la question de son remplacement.
Le Parlement Européen a récemment développé son nouveau système d'information : *Arpege*. Ce dernier est susceptible d'être choisi par le Conseil des Ministres.
L'étude qui suit, tente dans un premier temps d'exposer les deux systèmes et, dans un deuxième temps, d'intégrer Arpege au Conseil.



## ABSTRACT

The present Information system of the Council of the Ministers of the European Communities have now reached its term.
That's why, the analysts of the Computer Department is thinking about his replacement.
The European Parliament has recently developped its new information system, called Arpege, which might be chosen by the Council of the Ministers.
The following report tries in a first stage to introduce and explain those two systems, and then tries to inport Arpege to the Council of the Ministers.

# REMERCIEMENTS

J'aimerais remercier ici les personnes qui m'ont apporté leur aide pour la bonne réalisation de ce rapport de stage.

Mes premiers remerciements s'adressent à **M. J. Fichefet**, promoteur de ce mémoire, qui m'a permis d'effectuer le stage.

De même, je remercie tout le service informatique du Conseil des Ministres CEE, particulièrement **Mr. B. Muller** et **Mr. J-M. Vandeputte** pour leur accueil et leurs conseils ainsi que pour le temps qu'ils m'ont consacré et l'expérience dont ils m'ont fait profiter.

Je tiens aussi à exprimer ma gratitude envers **M. F. Gabarron** et **Mlle J. Barth** du Parlement Européen (P.E.) à Luxembourg, pour les nombreuses entrevues qu'ils m'ont accordées dans le cadre de mon analyse d'Arpege.

Enfin, je remercie toutes les personnes du Secrétariat Général du Conseil, du département Informatique du Parlement Européen à Luxembourg, de l'institut d'informatique qui, de près ou de loin, ont contribué à la réalisation de cette étude.

# TABLE DES MATIERES

## PARTIE 3: Migration de la configuration actuelle du Conseil vers Arpege



## PARTIE 4: Développement d'une nouvelle fonctionnalité au sein de l'application Arpege

# INTRODUCTION

Les organisations conçoivent, réalisent et utilisent des systèmes d'informations (S.I.) pour satisfaire les besoins en informations engendrés par leurs comportements organisationnels. Or, une organisation vit, évolue. On comprend dès lors aisement que les systèmes d'informations qui la composent doivent suivre cette évolution.

Le S.I existant au Conseil des Ministres des Communautés Européennes a été conçu à l'image des besoins en informations nécessaires à l'époque de sa création. Depuis sa naissance, les besoins informationnels du Conseil ont changés. Certes, on a adapté le S.I. (en y greffant de nouvelles fonctionnalités), mais on n'a pas pu changer, de manière significative, son architecture (tant logique que physique).
Ceci pourrait être une explication à la gestion "rudimentaire" des postes et de l'historique au Conseil.

Un S.I. connait donc une naissance, une évolution et une mort. Dans une organisation, il arrive un moment où il devient préferable (si pas nécessaire) soit de concevoir un nouveau système, soit de le convertir tout en changeant radicalement son architecture.
C'est vers cette seconde hupothèse que le Conseil des Ministres CEE s'est tourné. Pour cela il a choisi d'adopter un nouveau S.I., celui du Parlement Européen.
Ce dernier ayant globalement les mêmes besoins informationnels que ceux du Conseil.

Le présent document est le fruit d'une étude concernant le remplacement du système d'Information du Conseil par le nouveau système développé au Parlement Européen (P.E.) à Luxembourg, ***ARPEGE.***[1]
Les objectifs poursuivis sont d'ordres divers.

Le document vise d'abord à fournir une connaissance suffisante des deux systèmes, respectivement, celui du Conseil (Partie 1 de ce mémoire ) et celui du P.E (Partie 2). L'accent est porté sur ce dernier car les personnes qui auront à développer ce projet connaissent le système du Conseil alors que leurs connaissances sur Arpege sont assez rudimentaires. D'autre part, étant donné qu'il faudra passer d'un système à l'autre il faut avoir un niveau de connaissances suffisantes de ces deux systèmes avant d'aborder le problème de la "migration".

---

(1) Système **A**dministratif de **R**enseignements sur le **P**ersonnel et la **G**estion des **E**mplois

Par conséquent, une bonne partie de l'analyse (Partie 2) sera consacrée à la présentation de l'environnement ARPEGE ainsi qu'a **sa philosophie de conception de manière à fournir une base de départ pour les approfondissements spécifiques au cours des phases futures de ce projet.**

Une fois que les deux systèmes exposés, on pourra dés lors réfléchir au *"comment"* passer du premier vers le second (Partie 3). Donner les scénarios possibles de migration de l'un vers l'autre. Mettre en relief les points de frictions probables et tenter d'y apporter des solutions.

Une autre partie de ce mémoire (Partie 4) sera consacrée aux problèmes rencontrés lors du developpement d'une nouvelle fonctionnalité au sein d'Arpege. L'objectif de cette partie sera de montrer comment se servir de tous les éléments constituant l'application Arpege, afin de pouvoir y intégrer une nouvelle fonctionnalité.

Enfin le présent document est accompagné d'annexes qui éclaireront le texte du mémoire.

**En conclusion, ce document se veut être une base de connaissances, de réflexions, afin de servir de point de départ pour les phases de développements futurs du projet ARPEGE au Conseil.**

# PARTIE 1. LE SYSTEME D'INFORMATION (S.I.) DU CONSEIL.

## 1. Introduction

L'objectif principal du service informatique du Conseil est, tout d'abord, la gestion des informations concernant les fonctionnaires et, en deuxième lieu, la fourniture d'un certain nombre de services à ses utilisateurs. Il n'effectue donc que des tâches purement administratives. Par conséquent toutes les données que ce service manipule concernent les fonctionnaires et leur environnement. En d'autres termes, le service informatique, afin de répondre aux besoins des autres services, élabore des applications (infra point 3) qui vont manipuler un ensemble d'informations (infra point 2).
Nous allons, sans trop rentrer dans les détails, ce système d'information du Conseil.

## 2. Les données.

Les informations sont gérées par le sgdb(1) ISDAM de Siemens. Ces données sont regroupées principalement dans un seul fichier appelé **le Signaletic**.

### 2.1. Structure logique d'un fichier Isdam (le Signaletic).

L'enregistrement est un ensemble de zones, de longueurs variables.

| | | | | |
|---|---|---|---|---|
| Record 1 | L | zone1 | L | zone2 |
| Record 2 | L | zone1 | L | zone2 |
| | | | | |

Chaque zone est préfixée par une zone, L, qui indique sa longueur.

(1) Système de gestion de base de donnée, ou dbms (data base management system)

Il y a existence d'une clé principale qui identifie biunivoquement un enregistrement. Il y a de même la possibilité de créer des descripteurs (zones spécifiquement choisies dans chaque enregistrement en vue de recherches "complexes").
Les accès au fichier s'effectuent par:

-*la clé principale*: on accède alors à un enregistrement désiré.
-*un descripteur* : en donnant le nom et la valeur d'un descripteur, on accède à une liste de pointeurs vers les débuts des enregistrements répondant au critère de recherche.
-*une question complexe* : en formulant une question ( sur les valeurs des zones), on crée une liste de pointeurs qui est alors exploitée de façon analogue à l'accès via un descripteur.

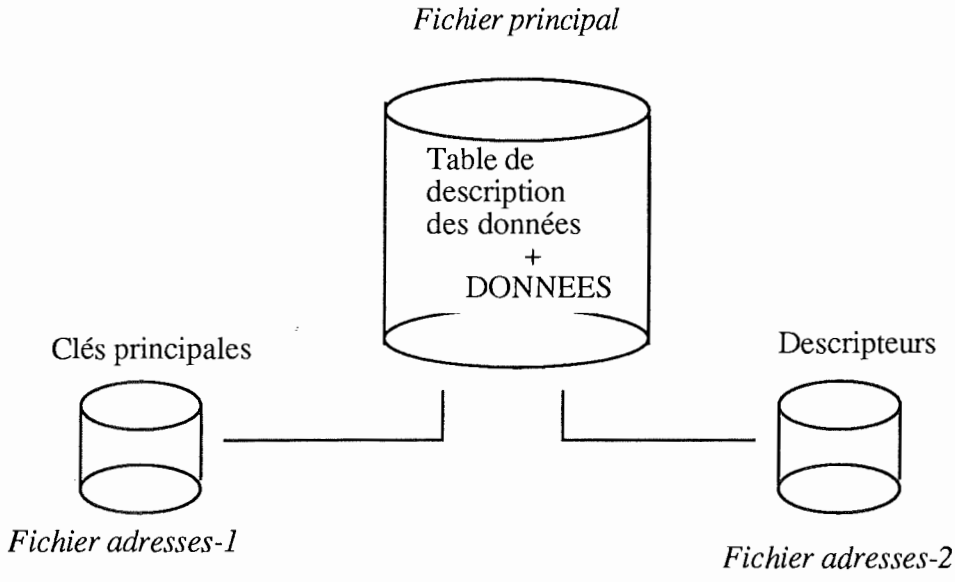
## 2.2. Structure physique du fichier Signaletic.

Les information sont stockées dans 3 fichiers physiques distincts:

-le fichier principal
-le fichier d'adresses-1
-le fichier d'adresses-2

### 2.2.1. Le fichier principal (le Signaletic)

Ce fichier comporte toutes les informations concernant les fonctionnaires. Il est scindé en deux. D'une part, la *table de description des données* et, d'autre part les *données* elles-mêmes.



-la *table de description des données* contient:

* le nom, la position, la longueur, le type de chaque donnée élémentaire pour chaque type de record prévu.
* des informations concernant les places libres dans la zone des données.

-la zone des *données*: contient toutes les informations concernant les fonctionnaires.

### 2.2.2. Structure du fichier d'adresses-1.

A chaque clé principale (numéro de matricule, par exemple) correspond un record du fichier d'adresses-1 qui contient cette clé, l'adresse de l'enregistrement logique correspondant.

### 2.2.3. Structure du fichier adresses-2.

Ce fichier contient les index permettant de retrouver tous les enregistrements qui contiennent l'argument de recherche utilisé (descripteur).

## 2.3. Les appels ISDAM.

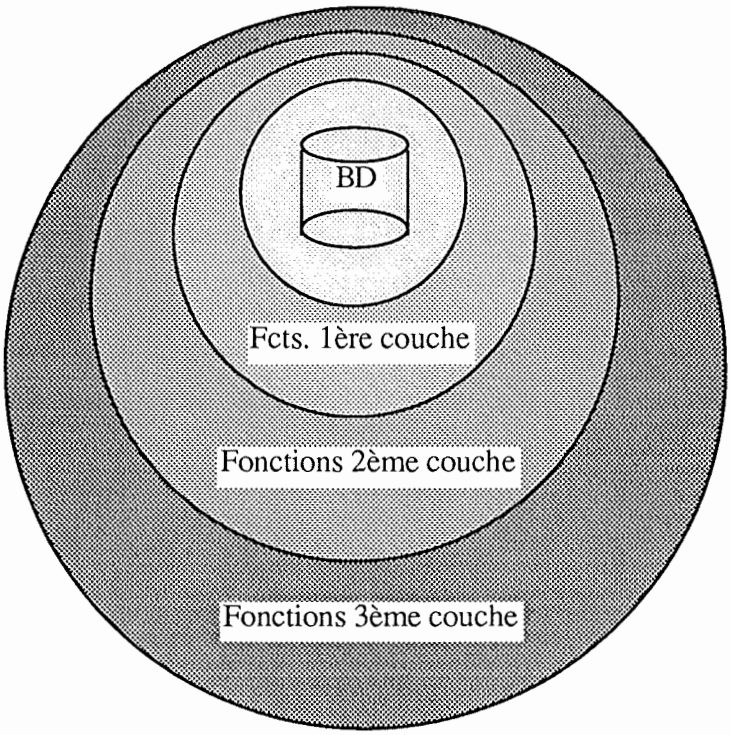
L'appel du sous programme de gestion ISDAM se fait à l'aide du verbe classique CALL. C'est ainsi que dans la plupart des applications, (généralement écrites en Cobol au Conseil) lorsqu'on veut accéder à un enregistrement, on le fait moyennant l'instruction:

```
CALL "PROG(1)" USING PARAM 1 PARAM2
```

# 3. Les applications.

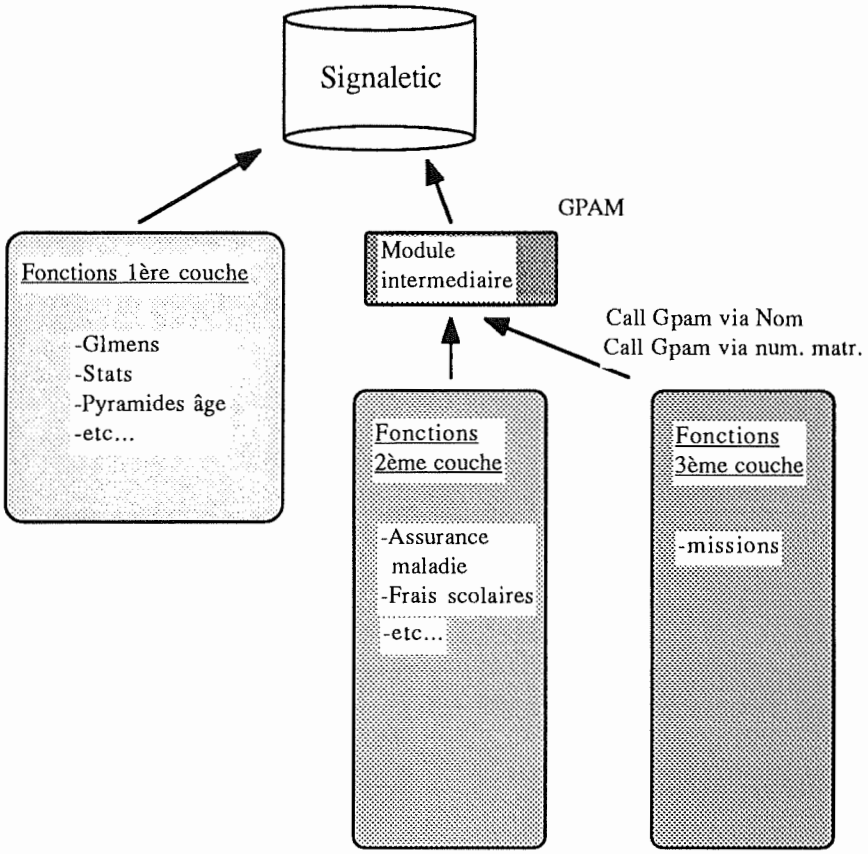
Afin de répondre aux besoins informationnels du Conseil, il existe un certain nombre d'applications. De manière à les distinguer on peut les regrouper comme suit:

**Classification des Applications du Conseil**




Les fonctions de la **première couche** sont articulées entièrement autour des informations contenues dans le Signaletic. Elles sont entièrement dépendantes de ce dernier et ne sont en interrelation avec aucune autre source de données ou paramètres. Le Signaletic et ces fonctions sont considérés comme le noyau du système.

Les fonctions de la seconde et troisième couche interagissent avec le Signaletic via le *module GPAM* (voir figure suivante) Ce dernier, ecrit en Assembler, se charge de fournir à ces applications l'enregistrement qu'ils recherchent (via un *buffer*) en fonction du *numéro de matricule* où du *Nom*..C'est ainsi que ces applications consultent, et parfois modifient, les enregistrements contenus dans le Signaletic.



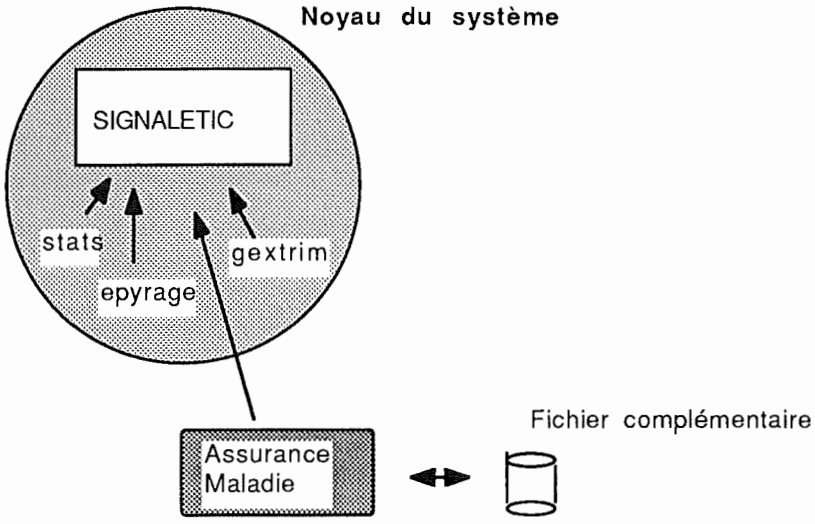

La liste, non exhaustive, des fonctions de la *première couche* est la suivante:

-GLMENS (l'une des plus importantes)[1]
-EPYRAGE
-EFFLIST1
-GSTATI7
-GSTATI8.B
-GSTATI8.A
-GIART
-GSTATI2
-GBUSEUR
-GREGIME
-GECHEAN
-GSTATI4

Contrairement aux fonctions de la *première couche,* **celles de la seconde** consultent les informations du Signaletic mais aussi les informations provenant d'autres fichiers (*fichiers complémentaires*) qui leurs sont propres.

---

[1] Au cours de la partie 4, il sera question de concevoir cette fonctionnalité.




La liste des applications *2ème couche* est:

-ASSURANCE MALADIE (A.M.)
-FRAIS SCOLAIRES
-VOYAGES ANNUELS
-PRIVILEGES IMMUNITES
-SERVICE SOCIAL (en cours de réalisation)

La dépendance de ces applications vis-à-vis du Signaletic est moindre que dans les fonctions de la *première couche*. Il sera question dans la nouvelle BD de prendre en charge tous ces fichiers complémentaires.

Les applications de la *troisième couche* consultent également le Signaletic, mais très ponctuellement, et sont en forte interrelation avec d'autres fichiers complémentaires. **Cette couche diffère de la seconde du fait que les fichiers complémentaires peuvent ne pas être pris en considération dans la BD Arpege.** Ceci est dû à la "distance" de ces applications vis-à-vis des données du Signaletic. Il s'avère, en effet, inutile d'intégrer dans la BD nouvelle un ensemble de fichiers complémentaires alors que les interrelations avec le Signaletic sont faibles.

La liste des applications de la *troisième couche* est la suivante:

-MISSION
-TRAITEMENTS
-COLIS ANNUELS
-COMPTABILITE

## 3. Les raisons de son remplacement par Arpege.

La principale raison de ce changement est le fait de l'arrêt de l'exploitation du système de gestion de base de données (SGBD) ISDAM. Le système d'information du Conseil ne pourra donc plus être exploité dans son état actuel et devra être adapté avant 1993, date de fin d'exploitation.

Par ailleurs le système, vieux de plus d'une dizaine d'années, devenait lourd à l'utilisation. Ceci résulte de ce qu'une organisation évolue et entraîne, par la même, l'évolution de son système d'Information (qui est à l'image des besoins informationnels de ses utilisateurs) de manière à répondre le plus souplement possible aux besoins provenant des services constituant l'organisation.

# PARTIE 2. LE SYSTEME D'INFORMATION DU P.E. A LUXEMBOURG : ARPEGE

## 1. Préliminaires.

Le système Arpege installé au P.E. constitue un ensemble de données et de fonctionnalités assez important. Au cours de cette partie, nous tenterons de présenter les différentes fonctions qui le caractérisent. Outre une description de l'utilité au point de vue *utilisateur* (*"partie visible de l'iceberg"*), nous nous efforcerons, lorsque cela est possible, de donner une description succincte, **au niveau conceptuel** de la fonctionnalité (*"partie cachée de l'iceberg"*). Ceci dans le but de fournir une **connaissance "conceptuelle" et "utilisatrice" d'Arpege nécessaire aux responsables qui auront plus tard à développer et maintenir le système au Conseil.**

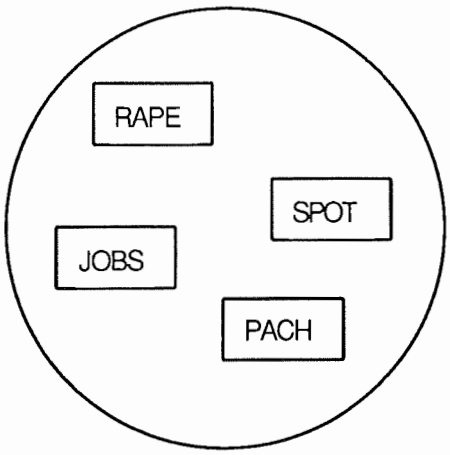
## 2. Configuration du S.I.(2) au P.E. avant l'introduction d'Arpege.

Afin de comprendre la naissance et l'évolution d'Arpege, penchons-nous d'abord sur son origine.

Le système en place au P.E. avant Arpege était constitué de quatre applications:

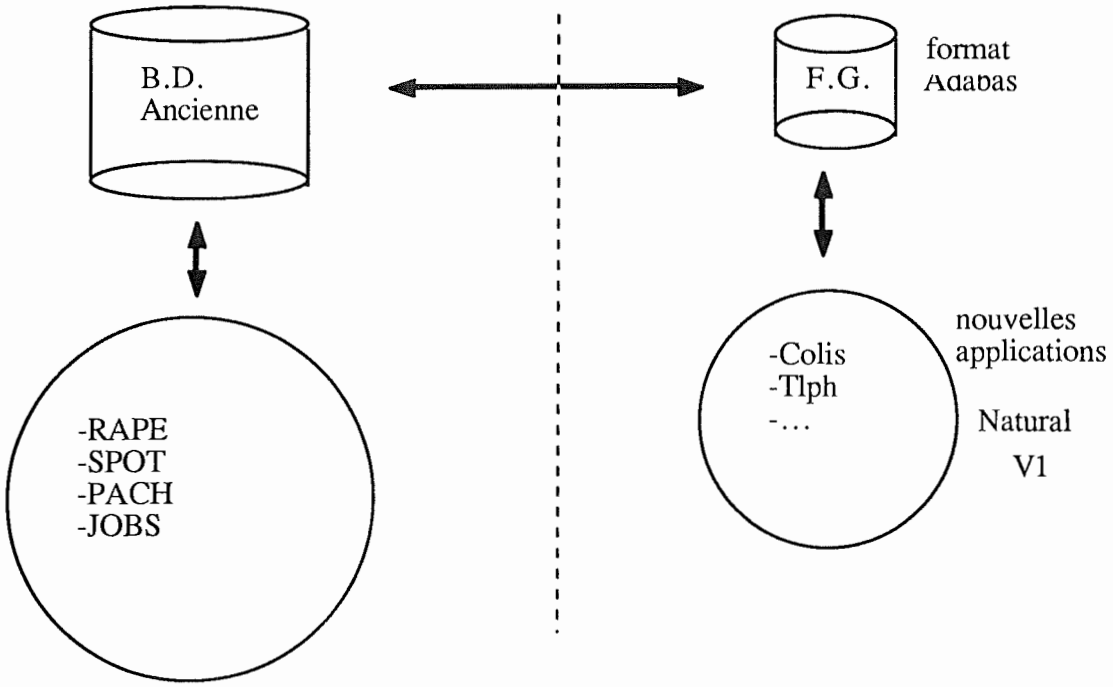
-RAPE (gestion des informations concernant le fonctionnaire)
-SPOT (gestion des postes)
-PACH (gestion des personnes à charge du fonctionnaire)
-JOBS (description des postes)

(2) Système d'Information. C'est l'ensemble des données et des applications qui composent un système informatique sans tenir compte du matériel ni du langage d'implementation de ces applications.

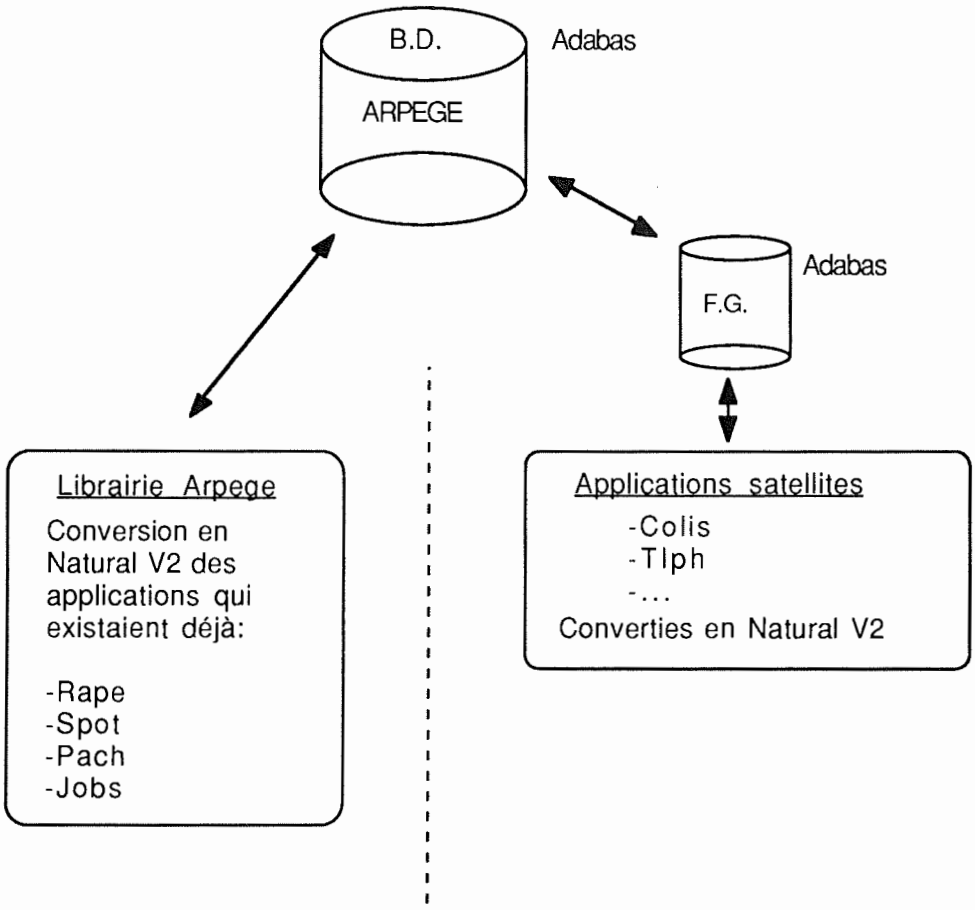



Au cours de la période de *gestation* d'Arpege, il est apparu que certaines applications nouvelles et "urgentes" devenaient nécessaires pour certains services. Il devenait impératif, pour le service informatique, de les développer en tenant compte du futur système Arpege (sgdb **Adabas** + langage de programmation **Natural**).

Le compromis suivant fût adopté: les applications furent developpées en Natural *version 1* et les données associées (en attendant la mise sur pied de la BD Arpege) ont été regroupées dans un fichier (fichier généralisé : **FG**) structuré sous le format Adabas.

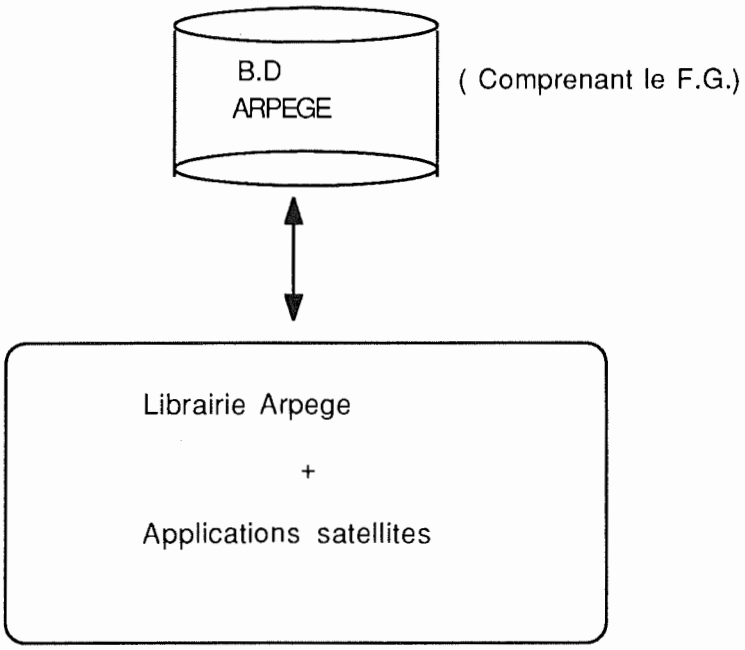


Lors de l'installation d'Adabas/Natural, où on venait de converti les anciennes applications (Rape, Spot, Pach, Jobs) ainsi que la BD Ancienne, la configuration était la suivante:



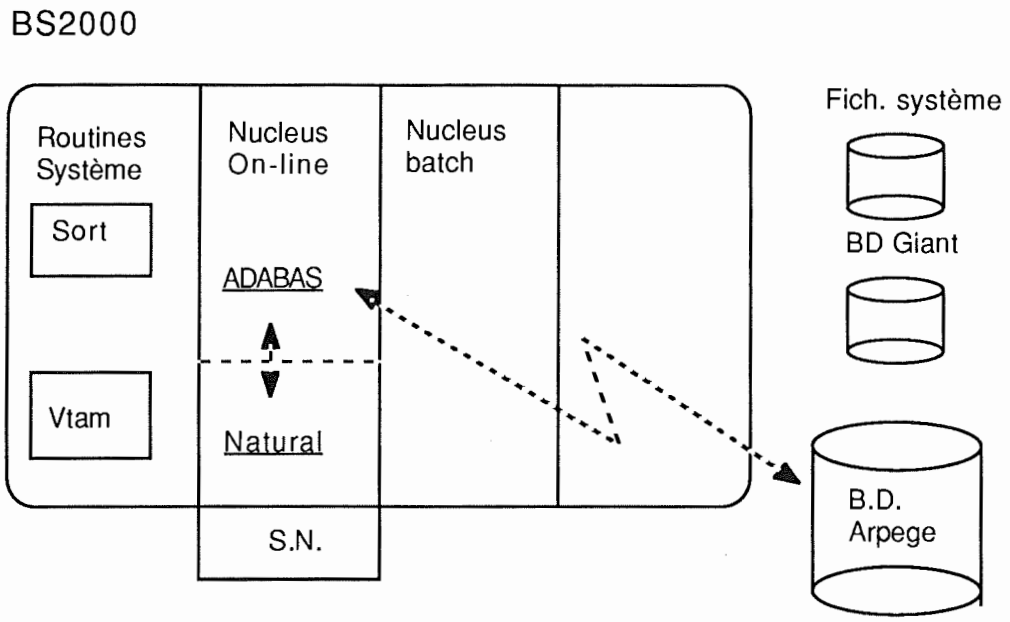

Dès l'apparition de la version 2 de Natural , les applications déjà développées en version 1 ont éte converties.

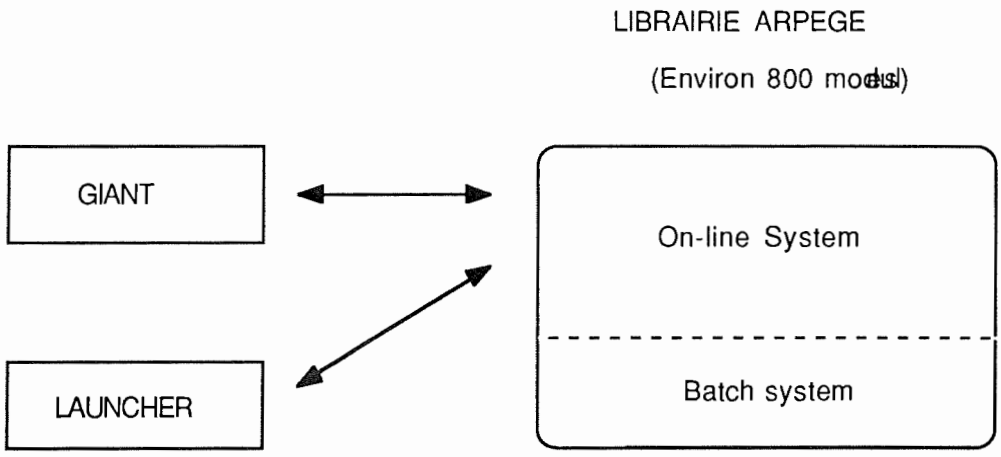
Au fil du temps, les application satellites seront directement intégrées dans Arpege (c'est le cas de *colis*, *missions* ),de sorte qu'à long terme, l'absorption de toute la structure satellite (FG + applications satellites) au sein d'Arpege permettra de constituer un seul ensemble homogène.




## 3. Architecture actuelle de l'application Arpege.

La configuration actuelle du système Arpege se **caractérisé par une distinction entre le "niveau système" (point 3.1.) et le "niveau applications" (point 3.2.)**. Le premier concerne le système d'exploitation BS2000 de Siemens, et le second les applications développées par le département informatique du P.E.




### 3.1. Le point de vue "système".

Sous le système du BS2000 existent différents "Nucleus" ou "directories".

#### 1) Nucleus Adabas/Natural:

Lors de la décision de développer Arpege, on a fait installer les softwares nécessaires à cette exploitation, c'est-à-dire:

## A) ADABAS:

C'est un système de gestion de bases de données (SGBD). Son rôle est de gérer les accès aux fichiers dans la B.D. Arpege. Chaque fois qu'une application demande des informations contenues dans la B.D. Arpege, c'est au SGBD qu'elle devra formuler sa demande. En fonction de critères de selection, d'adresses logiques, d'adresses physiques, de tables, ... ce SGBD ira placer le record demandé dans un tampon ( à une adresse connue) où l'application pourra aller chercher ses informations (on verra, dans le point 4 de cette même partie, comment le Sgbd organise et gère ses données).
Le SGBD est utilisé par Natural via des requêtes d'accès à un record d'un fichier de la B.D. d'Arpege (*Read, Write* un buffer). C'est le SGBD qui sait, en fonction du nom du fichier, où est situé physiquement le fichier (c'est-à-dire dans quel disque, à quelle partition, ...).
A ce titre voir infra 4.4.

## B) NATURAL (version 2)

C'est un langage de programmation au moyen duquel ont été développés tous les modules, **environ 800**, contenus dans la librairie Arpege. Il interagit à plusieurs niveaux:

- pour les I/O (inputs/outputs): il fait appel à Adabas afin d'obtenir le record désiré se trouvant dans un fichier précisé (cela via un tampon).

- pour les interactions avec l'utilisateur: c'est-à-dire, les demandes d'affichages, de saisies ..., à l'écran, il fait appel à des routines du système (VTAM) via les "datacom" (DC).

Ce langage impose une logique, une façon de travailler (de programmer) différente de celle en Cobol. Il possède néanmoins un rapport de productivité supérieur à ce dernier.
On verra au cours de la Partie 4 comment developper une fonctionnalité au moyen de ce langage de commande.

### C) S.N. (SUPER NATURAL).

C'est un langage de programmation évolué mis principalement à la disposition des utilisateurs afin qu'ils puissent répondre eux-mêmes à leurs propres besoins informationnels.
Il permet:

-l'extraction de données d'un, ou de plusieurs fichiers, selon un certain critère.

-outre l'extraction, il permet d'effectuer des opérations sur ces données.

A cette fin il convient de:

-connaître exactement les résultats souhaités avant toute réalisation du programme,

-savoir où (dans quels fichiers) trouver les zones d'intérêt,

-savoir définir exactement le critère de sélection,

-bien maîtriser les méthodes qui permettent d'aboutir aux résultats, sans pour autant être un bon programmeur.

On verra dans la Partie 4 comment utiliser cet outil pour réaliser une fonctionnalité.

Cet outil constitue donc un moyen d'utilisation aisée permettant aux services de créer eux même leurs propres listes (du même coup, le service Informatique se décharge de ces problèmes spécifiques).

### D) AUTRES OUTILS.

Il existe un utilitaire au sein d'Arpege qui permet l'extraction de données en vue de leur utilisation et de leur exploitation, dans un environnement PC, par des logiciels tels que DBASE 3 +, ou autres.
Cette extraction se concrétise par la création d'un fichier Ascii qui sera lu par le logiciel sur le PC.

## 2) Nucleus routines du système:

Contient toutes les routines utilitaires contenues dans tout BS2000, c'est-à-dire, sort, merge, ...

### 3) Nucleus batch:

Une autre distinction à faire concerne les opérations appelées transactionelles (*on-line*) et les opérations *batch*. Les premières portent sur un seul record à la fois (transactionnelles), les deuxièmes sur un ensemble de records. L'ordinateur étant une ressource fortement demandée, il faut parfois postposer des travaux moins urgents de manière à donner la priorité à certains travaux plus "pressants".
C'est ainsi que les opérations transactionnelles ont généralement la priorité sur les opérations opérations batch. Le nucleus batch se chargeant de la prise en compte des travaux batch. On verra que, dans le niveau applications, la tâche du *launcher* consiste à gérer le lancement et le suivi des programmes batch.

### 4) Les données.

**a) La B.D. d'Arpege** contient les informations essentielles concernant la gestion de personnel. Elle comporte principalement **cinq fichiers** :

-fichier Personnel (FG-PERS)
-fichier Emplois (FG-POSTE)
-fichier Famille (FG-FAMI)
-fichier Historique (FG-HISTORY)
-fichier des tables "privées" à ces quatre fichiers. Par exemple: code pays, état civil... (FG-TABLES)

Voir Annexes 2 la liste compléte des champs des trois premiers fichiers *("Nomenclature Arpege")*

### b) La B.D. Giant.

Contient des données propres au logiciel Giant. Ce dernier est une librairie de programmes qui interagissent entre eux pour fournir les fonctionnalités attendues. Giant s'occupe aussi de la sécurité (particulièrement les *"Entry-points"*) et doit garder certaines informations utiles à cette fin. Exemple de tables: via Giant on peut demander directement l'exécution d'un programme en l'appelant de diverses manières.

Ainsi, par exemple, un programme "gestion de la personne" peut être appelé "gestpe" ou "gp". C'est alors à Giant de comprendre que l'utilisateur veut exécuter le programme "gestion de la personne" et donc le programme numéro 50 par exemple. Bien évidemment, le nombre de synonymes différents pour un même programme est limité.

Ces données sont contenues dans la "table des mnémoniques" qui se trouve dans la B.D. Giant.

### 3.2. Au point de vue applications.

La conversion des anciennes applications RAPE, SPOT, JOBS, PACH, a donné naissance aux modules ou programmes (environ 800) contenus dans la librairie des applications Arpege, **laquelle librairie constitue l'application Arpege**[1]. Ces modules ont été écrits, où plus exactement convertis, en Natural V2. La librairie est une liste de modules qui interagissent entre eux et qui reprennent toutes (et même davantage) les fonctionnalités existantes dans les quatre applications anciennes. On remarque la distinction entre la librairie de programmes *on-line* et la librairie de programmes *batch*. Les modules de la première travaillent de manière transactionnelle. Ceux de la seconde visent un ensemble de records à la fois. On verra plus loin (point 8) la liste des programmes batch qui sont contenus dans cette librairie. Ces modules utilisent des outils considérés comme standards pour toutes les applications. On a voulu par là "factoriser" le travail. C'est le cas de *Giant* et du *Launcher*.

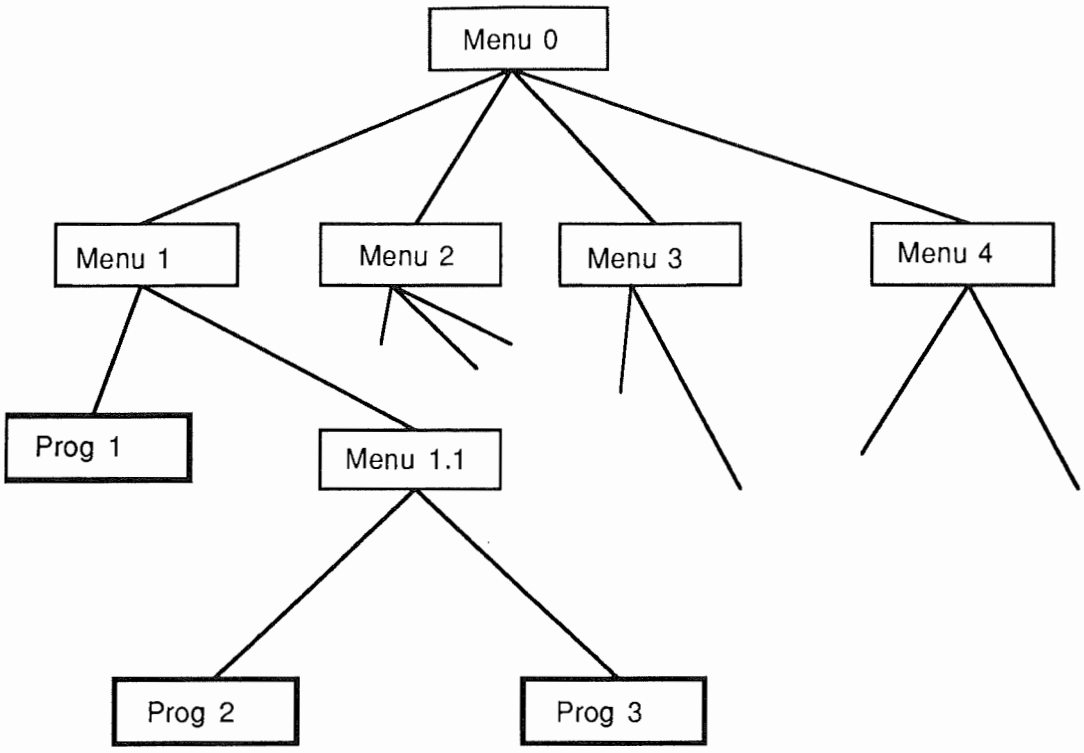
### 3.3. Les outils.

A) GIANT :

C'est une application assez importante qui permet la navigation au sein des menus et des programmes contenus dans la librairie d'Arpege. La navigation est rendue possible par la structuration de l'ensemble sous une forme arborescente:

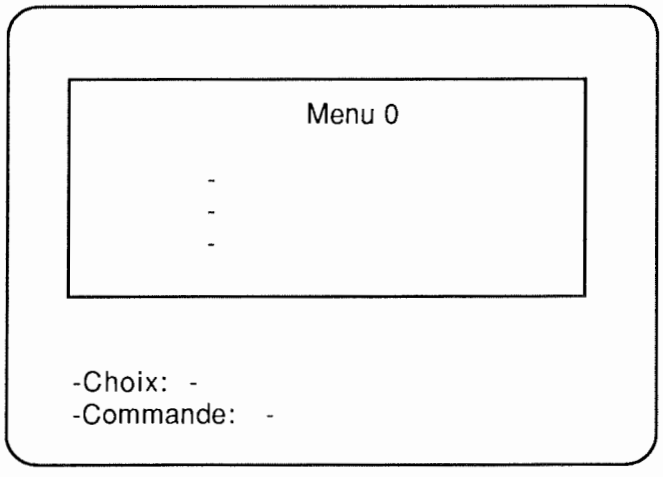
---

(1) Faire la distinction avec le Système Arpege qui est constitué de la BD Arpege avec l'application Arpege.




Ainsi, lorsqu'à partir de l'ecran de présentation (menu 0)

**Ecran de GIANT**




on veut executer le programme 2, on a le choix entre les possibilités suivantes:

- soit le faire niveau par niveau:

  dans "choix" : répondre 1 ensuite

  "choix": répondre 1.1 ensuite

  "choix": répondre prog 2

- soit l'exécuter directement avec la possibilité d'utiliser des mnémoniques (voir supra).

  Pour cela: dans "Commande": répondre prog 2 (ou "gestemplois" ou...d'autres mnémoniques)

**Ce produit** :

- est un produit réalisé par le P.E. en Natural. Il a été conçu pour factoriser le travail de navigation au sein d'Arpege et pour traiter le problème de la sécurité.

- gère les accès aux différentes branches de l'arbre Arpege. En fonction du *Userid*(1) , il donne accès à tout l'arbre ou à une de ses parties (à partir du menu 1.1. par exemple. Ainsi l'utilisateur ne pourra pas exécuter le Programme 1 ).

- gère la sécurité dans les accès aux informations. En effet, en plus de donner la possibilité de verrouiller certaines branches de l'arbre Arpege, il offre aussi la possibilité de verrouiller certains champs pour ainsi donner accès à tel ou tel champ en fonction du *Userid.*

B) Le LAUNCHER.

Ensemble de programmes qui permettent de gérer les demandes d'executions de programmes batch à partir des programmes *on-line.* C'est lui qui se charge de la gestion de la file d'attente des programmes batch à exécuter dans le nucleus Batch du BS2000. On verra dans la Partie 4 un exemple d'utilisation de ce Launcher.

(1) User-identification. A chaque user-identification correspond un niveau de securité caractérisant l'utilisateur.

## 4. Architecture B.D. Diffusion Information.

Dans un premier temps nous analyserons l'architecture logique et physique des données (respectivement point 4.1. et 4.2.) et ensuite nous exposerons la diffusion des informations au sein du P.E (point 4.3.).

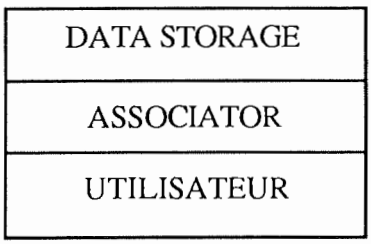
### 4.1. Architecture logique de la BD Arpege.

#### 4.1.1. Description du modèle.

Adabas comporte deux parties physiques distinctes:

-la DATA STORAGE: contenant toutes les données concernant les fonctionnaires
-l'ASSOCIATOR: contenant toutes les données stockées dans la DATA STORAGE.

Le système ne travaille pratiquement jamais directement sur les données vraies (Data Storage). Il passera toujours par un ensemble d'outils technologiques (Associator) représentant presque toujours les adresses de la DATA STORAGE ou sont stockées les données. En réalité le système fera constamment de l'adressage indirect.

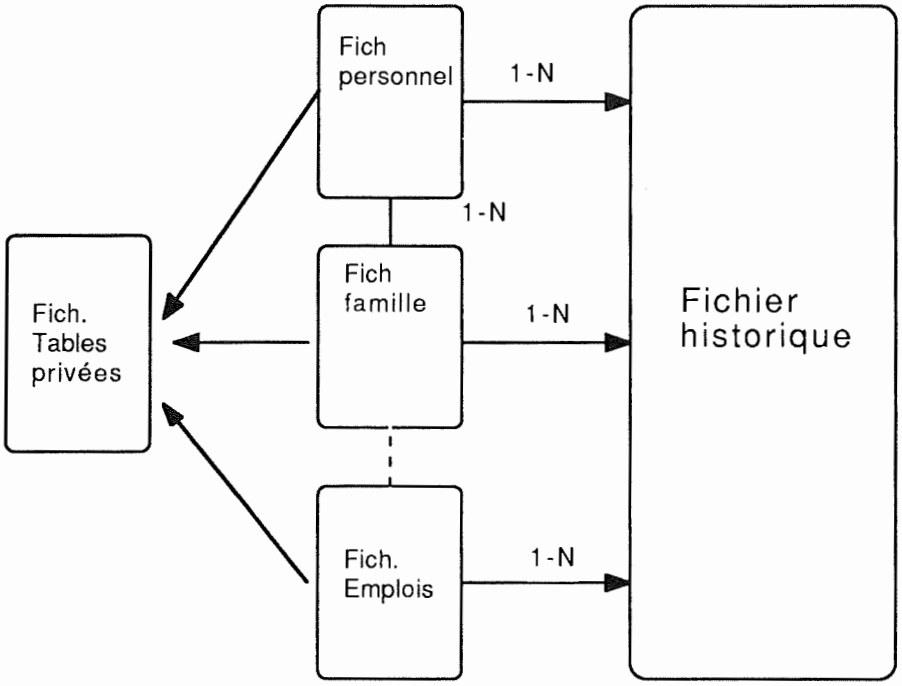



Exemple: si un utilisateur recherche tous les fonctionnaires ayant plus de 30 ans d'ancienneté pour leur allouer une prime, le système aura stocké dans l'associator (suite à un certain traitement de l'administrateur) toutes les adresses de la DATA STORAGE des personnes répondant au critère. Ce n'est qu'a partir de ces adresses que le système retrouvera les données vraies dans la DATA STORAGE.

#### 4.1.2. Représentation logique des enregistrements.

Dans Adabas une zone (ou un champs) s'appelle ELEMENTARY FIELD. Un ensemble de ELEMENTARY FIELD forme un enregistrement logique, et un ensemble d'enregistrements logiques de même structure forme un fichier logique.

Les fichiers constituant la BD Arpege sont articulés de la manière suivante:




Il existe trois fichiers (Emplois, Personnel, Personnes à charge) qui décrivent pour une personne la situation actuelle. Le fichier historique est unique pour tous les trois fichiers. Les connectivités 1-N signifient qu'à un enregistrement du premier fichier peut lui correspondre un ou plusieurs enregistrements du second. Ainsi pour une personne (du fichier personnel) peuvent lui correspondre plusieurs enregistrements historiques le concernant.

Un ELEMENTARY FIELD peut, à l'intérieur d'un enregistrement, se situer dans plusieurs ensembles différents:

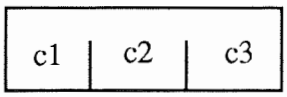
elle peut:

-être une donnée simple

| A | B |
|---|---|

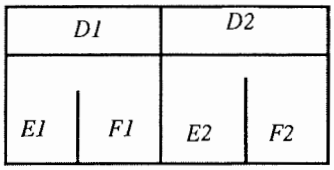
-représenter plusieurs valeurs successives d'une même donnée (zone *occurs* premier niveau en Cobol). C'est dans ce cas un MULTIPLE FIELD (C1 plus C2 plus C3).

*elementary field*




-être dans un ensemble de données répétitives (zone *occurs* à plusieurs niveaux). C'est dans ce cas un PERIODIC GROUP.

*Periodic group*

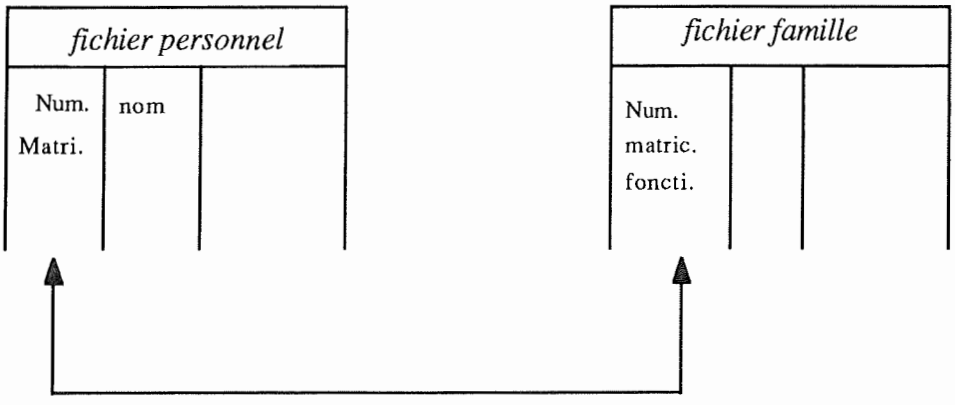



Les ensembles de données élémentaires (E1,F1), (E2,F2) forment alors un PERIODIC FIELD.

Dans chaque enregistrement d'un fichier, la clé d'accès s'appelle DESCRIPTOR FIELD. C'est sur ce dernier que sera construit, de manière automatique par le système, une liste inverse (ou liste de pointeurs).

Tout ELEMENTARY FIELD peut être un descripteur, ou, en d'autres termes, une clé d'accès. Il peut donc y avoir plusieurs descripteurs pour un même enregistrement.
L'unité d'accès logique est le champ ou FIELD (ELEMENTARY FIELD, MULTIPLE FIELD, PERIODIC GROUP).

Toute recherche dans Adabas passe par un descripteur (on verra ultérieurement comment cela se traduit physiquement).
Aussi toute relation entre les fichiers logiques ne peut-elle être qu'une relation entre les descripteurs. C'est ainsi qu'une relation entre des données appartenant à des fichiers logiques différents se traduiront par le COUPLAGE de deux descripteurs communs.




Lien de couplage

## 4.2. Architecture physique de la BD Arpege.

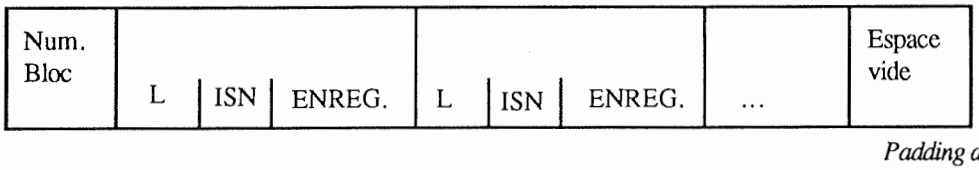
Comme dit précédemment, Adabas se compose de deux parties distinctes:

1. La DATA STORAGE où sont stockées physiquement toutes les données du système. Elles sont sous forme d'enregistrements et de fichiers classiques.

2. L'ASSOCIATOR contenant quant à lui toutes les données technologiques permettant d'effectuer toutes les opérations possibles sur les données vraies de la DATA STORAGE. Ces dernières sont stockées dans des blocs physiques de longueur fixe, par contre les enregistrements peuvent être de longueur variable.

### 4.2.1. LA DATA STORAGE.

La DATA STORAGE est l'espace physique où sont stockés les enregistrements. Chaque enregistrement validé (de manière a respecter l'intégrité de la BD) est stocké et reçoit un numéro d'ordre interne appèllé le *Internal Sequence Number* (ISN).
Il existe une possibilité de compressage des données. C'est-à-dire que chaque champ d'un enregistrement peut être compressé (suppression des espaces et caractères non significatifs).
Un bloc physique peut être représenté de la manière suivante:

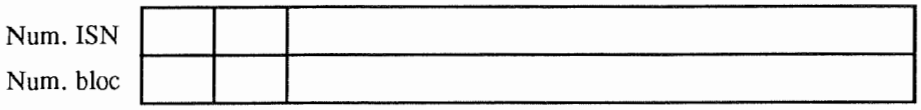



### 4.2.2. L'ASSOCIATOR.

C'est l'espace physique où sont stockées toutes les données techniques. Il comprend:

*(1) l'ADDRESS CONVERTER* (convertisseur d'adresses)

Crée une relation entre le numéro de l'ISN et le numéro du bloc



(2) *les INVERTED LISTS* (les listes inverses)

Pour chaque champ déclaré comme clé d'accès (DESCRIPTOR FIELD), ADABAS crée une liste inverse. Celle-ci est constituée de tous les numéros d'ISN qui répondent à une valeur du DESCRIPTOR FIELD.

Exemple:

*Fichier personnel*

| Matric. | Nom | ... | ISN |
|---|---|---|---|
| 01 | Dupond | | 100 |
| 02 | Durand | | 101 |
| 03 | Dupond | | 102 |
| 04 | Tartempion | | 103 |

Si on décide de créer un descripteur sur le nom on aura comme liste inversée:

| NOM | ISN |
|---|---|
| Dupont | 100, 102 |
| Durand | 103 |
| Tartempion | 101 |

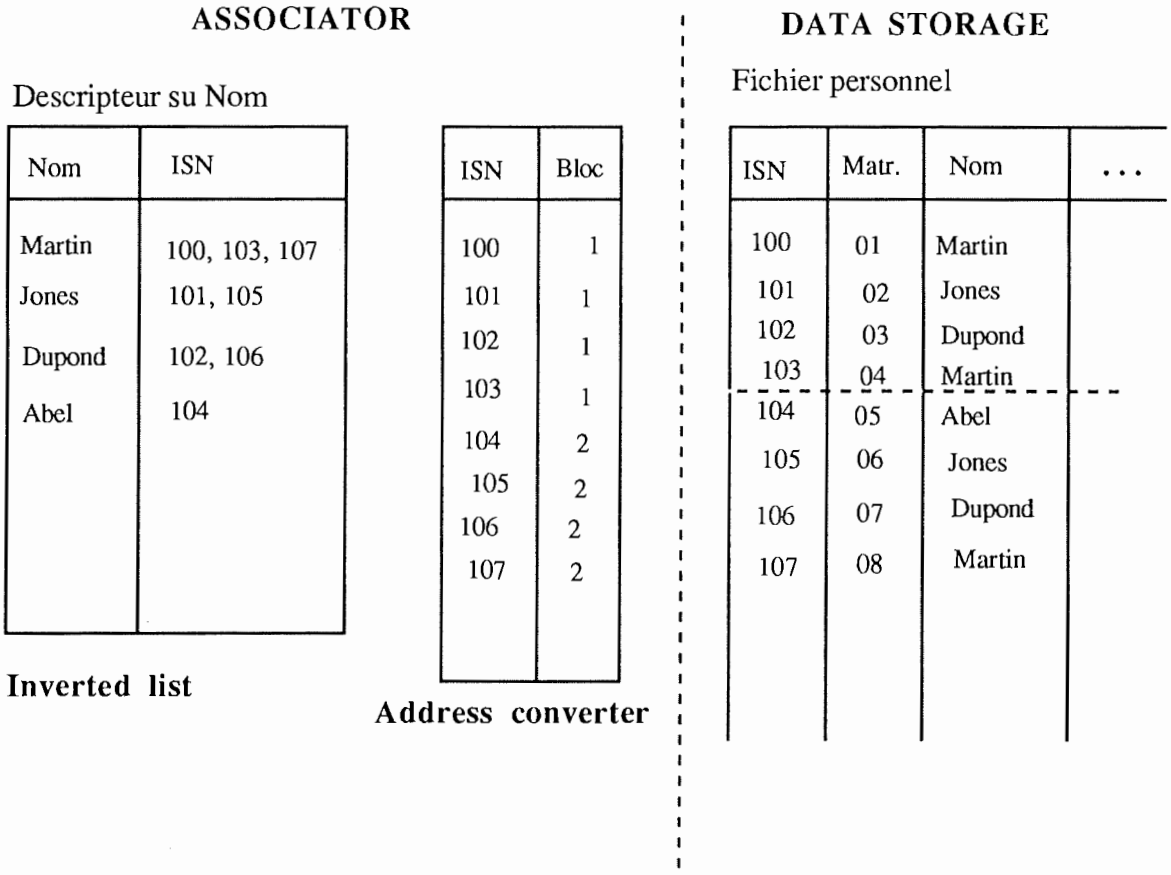
Si on décide de le créer sur Numéro de matricule, on aura comme liste inverse:

| Matricule | ISN |
|---|---|
| 01 | 100 |
| 02 | 101 |
| 03 | 102 |
| 04 | 103 |

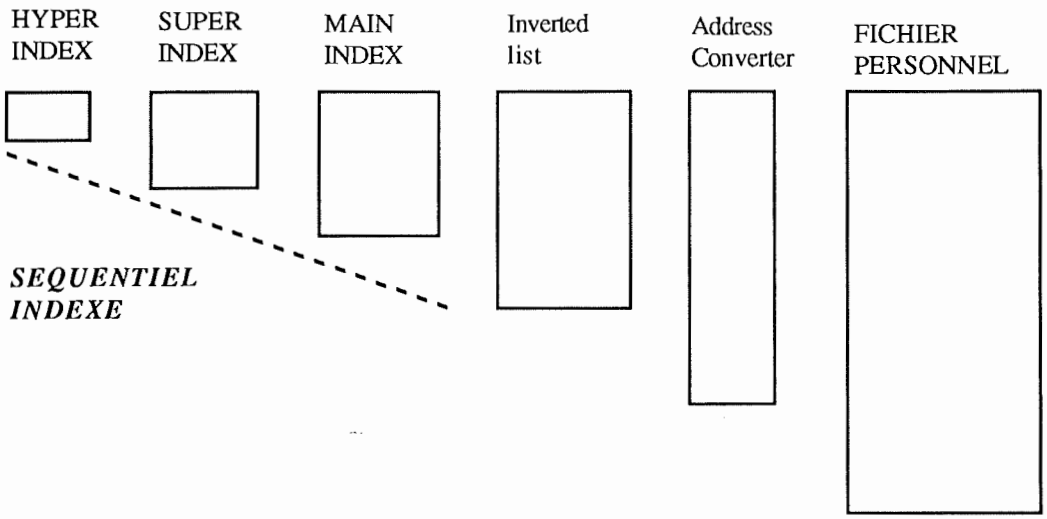
Si le champ est *identifiant* de l'enregistrement, alors on a un numéro d'ISN pour toute valeur de ce champ.

(3) différentes tables notamment les tables décrivant les différents champs (Field) et permettant la gestion de l'espace disponible dans les blocs.

### 4.2.3. L'interrelation DATA STORAGE et ASSOCIATOR.




La gestion de l'Inverted list est organisée en sequentiel indéxé.



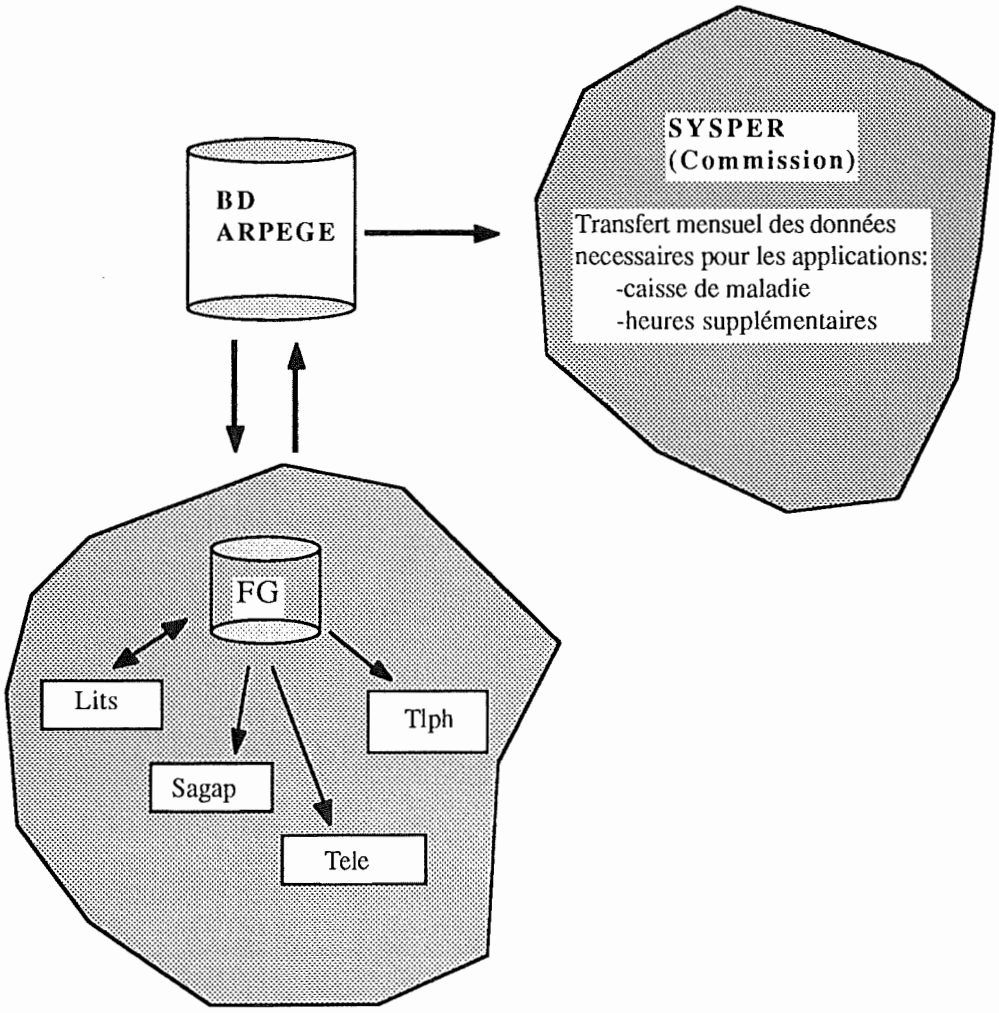

Lorsqu'une recherche s'effectue à partir de descripteurs, Adabas exploite les listes inverses pour retrouver les N° de ISN corréspondant, puis à partir de "l'Address converter" le système retrouve les différentes adresses physiques des enregistrements concernés. Mais avant cela, pour atteindre les listes inverses, il va explorer une table d'index à trois niveaux: Hyper, Super et Main Index.

### 4.2.4. Représentation concrète des relations.

La représentation de couplages entre deux fichiers F1 et F2 se conçoit aisément au moyen de deux listes inverses. La première pour effectuer la liaison de F1 vers F2, c'est-à-dire, à partir d'une valeur du descripteur de F1, quels sont les N° ISN du fichier F2 qui sont concernés. La deuxième liste fait le même travail à l'exception que le lien se fait du fichier F1 vers le fichier F2.

### 4.3. La diffusion de l'information.

A partir de la BD Arpege les informations sont distribuées comme suit:




Le fichier généralisé (FG), est une base de données locales dans laquelle les applications vont lire et gérer les informations qui leur sont necessaires. Ces applications satellites ont aussi leurs propres fichiers "privés".

Le transfert BD Arpege ---> FG, est un transfert journalier des informations relatives à la situation finale du personnel. Par conséquent, ces applications satellites travaillent avec les données de la journée précédente.
Le transfert FG ---> BD Arpege, est un transfert mensuel qui ne concerne que la gestion des listes téléphoniques (Lits)

En ce qui concerne le Conseil, seule la diffusion **Arpege ---> Sysper** nous intéresse puisqu'il doit aussi fournir des informations similaires à ce système de la Commission.

Petit à petit, le P.E. intègre les "applications satellites" au sein du système Arpege. Ce qui a comme conséquence, de restreindre la nécessité du F.G., puisqu'à long terme, il sera absorbé dans la B.D. Arpege.

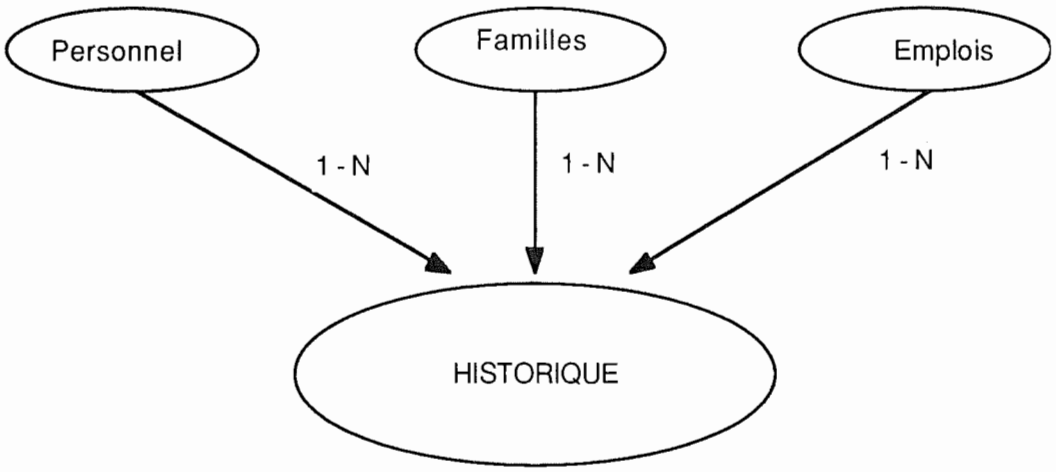
## 5. Le traitement de l'HISTORIQUE dans Arpege

Dans une organisation, les informations concernant une personne évoluent, changent. Il est indispensable pour les gestionnaires de ces personnes de connaître tous les changements de la situation d'un fonctionnaire dans le temps. En effet, si on veut connaître l'évolution de la carrière d'un fonctionnaireil faudra historiser tous les changements concernant sa carrière (catégorie, grade, échelon). Ainsi on peut reconstituer son évolution.

Il est question dans ce qui suit de présenter les fondements du concept d'historique sous Arpege. L'idée principale est de donner **les connaissances conceptuelles de base de la notion d'historique.** En ce qui concerne le point de vue purement technique, ce dernier ne sera pas traité en profondeur. Nous avons voulu donner par ce texte une connaissance de base (abstraite, théorique, conceptuelle) et non pas des approfondissements techniques.

Nous avons subdivisé ce problème en deux parties. La première tentera de décrire les concepts sous-jacents à la construction de cet historique (point 5.1.). La seconde décrira l'application ***Chronos*** (point 5.2.) qui prend en charge cette fonction d'historisation.

## 5.1. L'Historique du point de vue conceptuel.




L'historique du système est repris **entièrement dans un seul fichier.**

Chaque fichier possède un certain nombre de zones. Parmi ces zones certaines seront à historiser. Par exemple, pour le fichier personnel, on a environ 150 champs historisés sur un total d'approximativement 230 zones.

**Fichier personnel**

| Personnes | Hist. |
|---|---|
| Nom | Oui |
| Date nais | Non |
| Cat | Oui |

Puisqu'on veut garder trace de toute modification de chaque zone, pour chacune d'entre elle on aura plusieurs historisations distinctes par la "date d'historisation". Les

enregistrements historiques sont de type "atomique", c'est-à-dire, qu'un enregistrement concerne:

-un fichier (puisqu'il y en a plusieurs)
-une zone de ce fichier
-une date d'historisation (puisque pour une même zone on peut avoir plusieurs historisations).

Par conséquent un enregistrement de l'historique a comme identifiant (pas encore complet):

-file name
-field name
-date d'historisation

De plus, l'historique d'un fonctionnaire (c'est-à-dire d'un record du fichier personnel par exemple) devra être distinct de l'historique d'un autre fonctionnaire pour:

-un même fichier
-une même zone
-une même date d'historisation

Donc, afin de distinguer les fonctionnaires, on aura comme nouvel identifiant du fichier historique:

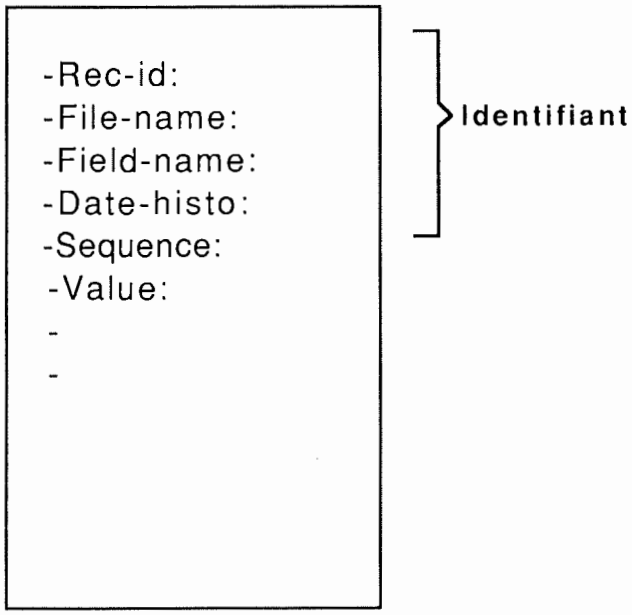
-record identification (Rec-Id.). Qui distingue les fonctionnaires. Peut, par exemple, être le numéro de matricule du fonctionnaire.
-file name
-field name
-date historisation

De plus, il porrait arriver que, pour le même fichier, pour la même zone, pour le même fonctionnaire et pour la même date d'historisation, on aît à effectuer plusieurs modifications. Il s'agirait alors de pouvoir effectuer plusieurs modifications d'une même zone le même jour.
Exemple : une personne qui change de numéro de téléphone deux fois en un jour.

Pour satisfaire à cette attente il faudrait ajouter dans l'identifiant minimal "la séquence" de toutes les modifications survenues le même jour. Cette séquence contient la

suite des modifications de la zone par ordre chronologique. La dernière étant la valeur actuelle de la zone.

Par conséquent, un record du fichier historique se compose, au minimum, des zones suivantes:




Dans la réalité, il s'avère que pour archiver une zone qui ne fait qu'un caractère de longueur (par exemple, l'état civil d'une personne), le record dans l'historique aura une longueur de 27 caractères (26 servant d'identifiant).

Peut être qu'une autre organisation de ce fichier historique donnerait un meilleur ratio

$$\frac{\text{taille effective de l'information}}{\text{taille totale record}} = \frac{1}{27} = \text{longueur totale du record historisé}$$

Cette politique de doter chaque record d'un bon identifiant, favorise le temps de réponse du système suite à une demande. En fait on y gagne en processing et on y perd en place mémoire. Mais le premier est plus intéressant que le second. Cet aspect des choses n'étant pas fort important puisque toute mémoire est extensible, la seule question qu'on pourrait se poser concerne les performances d'accès à un record de ce fichier historique.

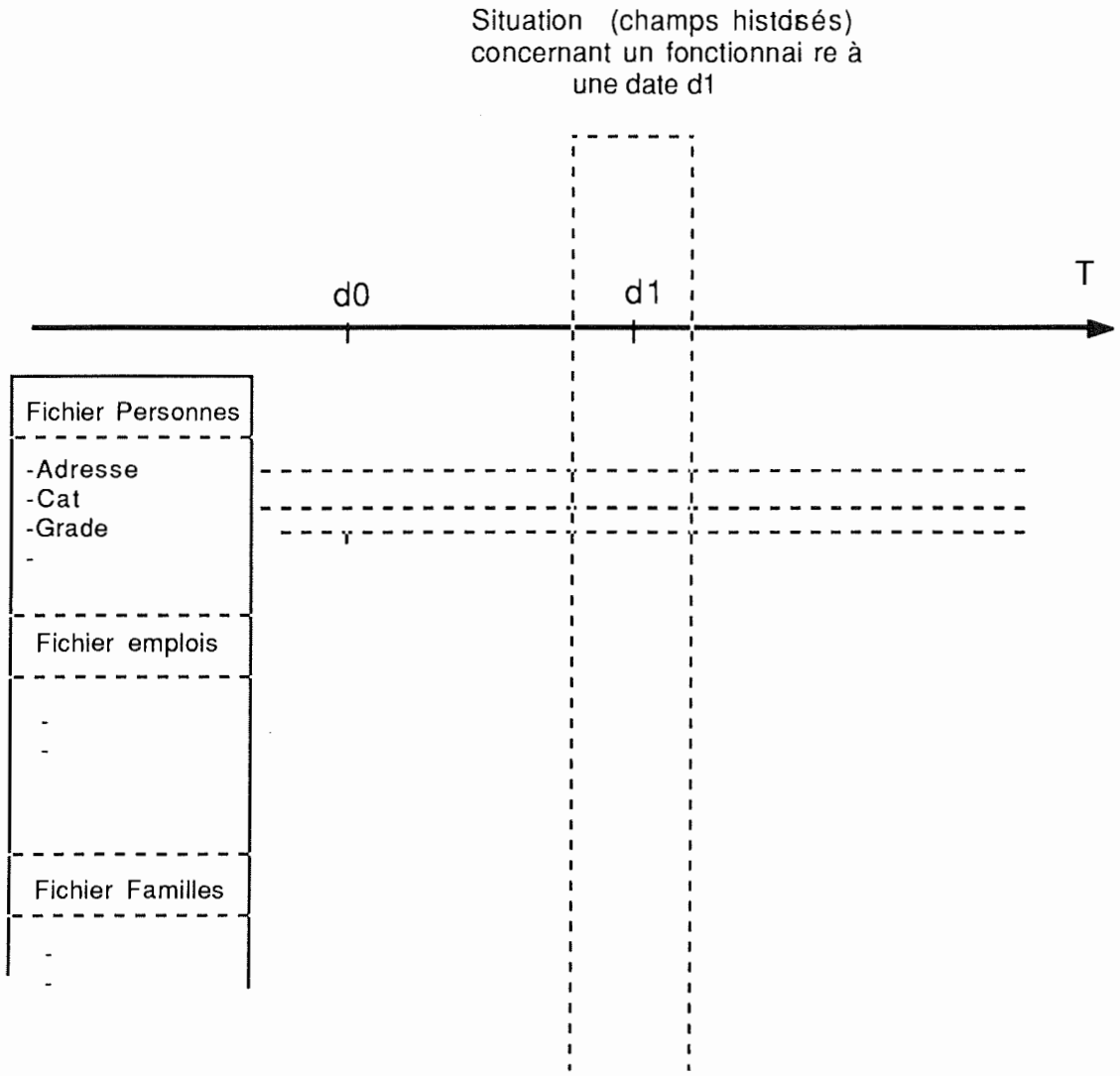
La vitesse d'obtention d'un enregistrement est assez rapide. Par exemple, toute opération de reconstitution de la situation totale d'une personne, à une date donnée (soit la

recherche des 150 champs historisés parmi les 880.000 constituants ce fichier Historique) prends un temps de réponse d'approximativement 3 secondes. Ce qui est assez acceptable lorsqu'on sait que ce fichier actuellement contient environ 880.000 records.

On pourrait se demander combien sera ce temps de réponse dans x années sachant que l'accroissement annuel du fichier est de 100.000 records.

En réalité, si ce temps se dégradait, il suffirait de segmenter (*de splitter*) ce fichier et ainsi d'en faire plusieurs. Par exemple, constituer un fichier historique pour chacun des trois fichiers concernés. Avant la recherche, il suffirait de demander à l'opérateur sur quel fichier porte la reconstitution.

De manière globale on peut schématiser le concept d'historique de la façon suivante:

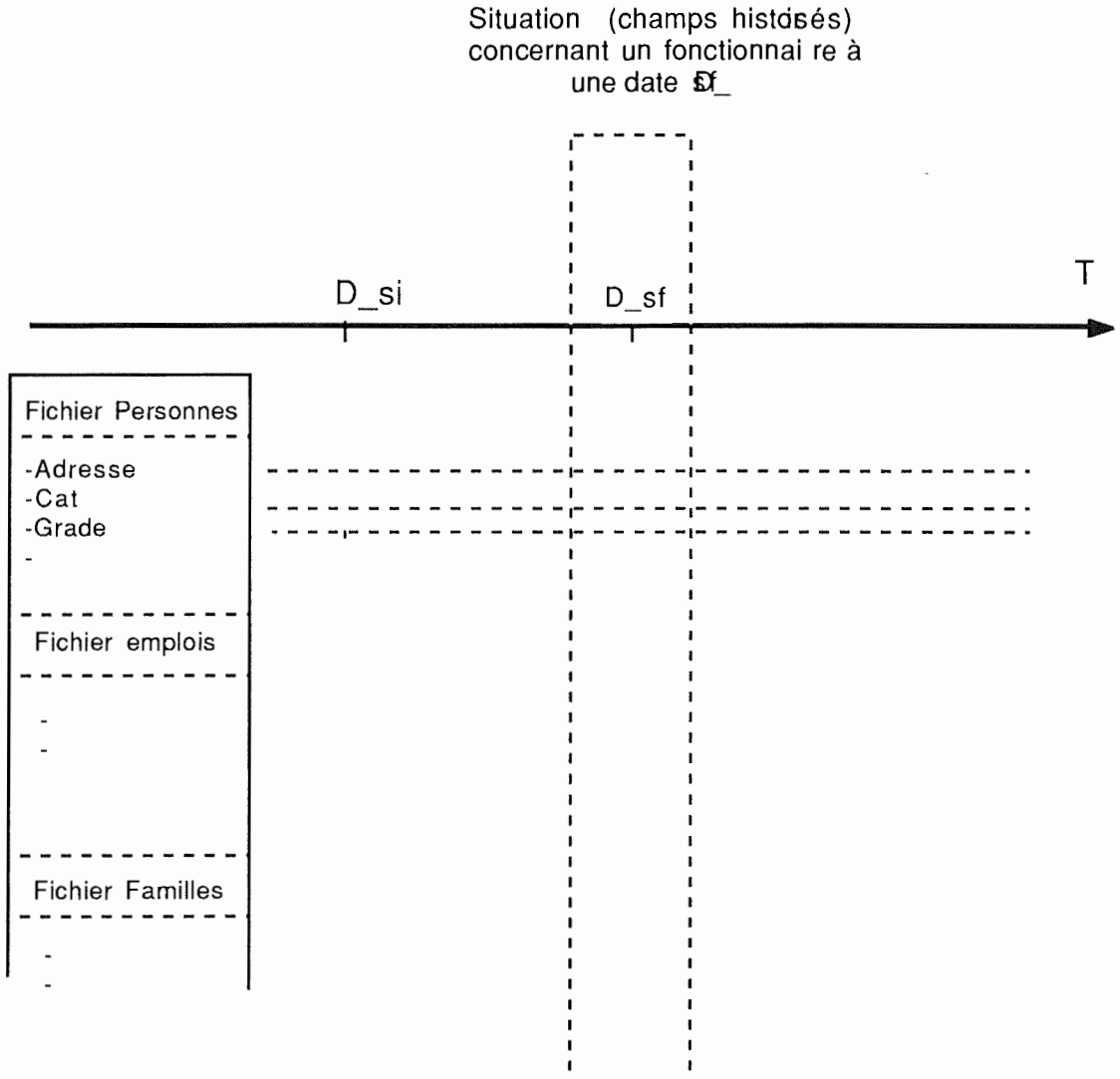


Chaque fichier possède un certain nombre de zones "historisables" reprises sous une date. Pour une modification d'un record (par exemple, changement d'adresse) donnant lieu à une nouvelle situation, il y a passage de la situation d0 à la situation d1. Cette historisation est prise en charge dans Arpege par une fonction appélée *Chronos*. Ce module historique se charge:

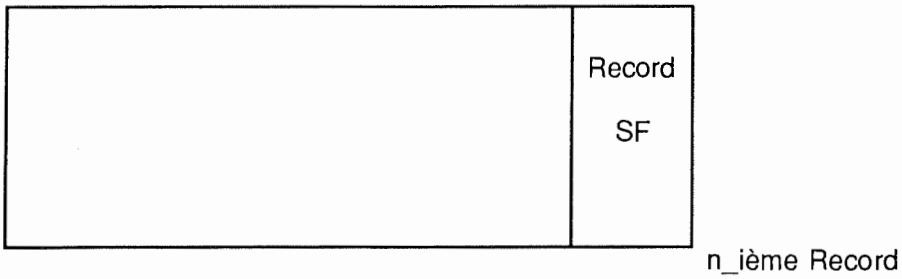
-des problèmes de cohérence à l'intérieur du fichier historique
-des champs à historiser (car toutes les modifications de champs ne sont pas historisées, seules celles concernant les champs historisables sont prises en charge).

**Les manipulations se partagent en deux types:**

1)Ajout d'historiques en cas de MAJ d'un enregistrement S.I. (situation finale).
Cas le plus fréquent.



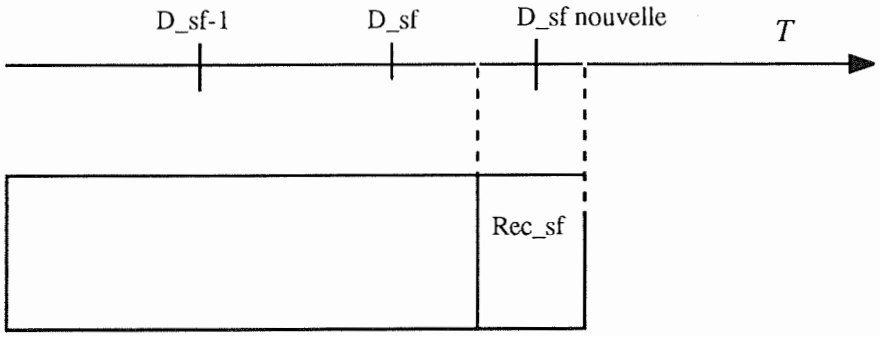

Fichier Personnes



Dans ce cas on veut modifier le Record SF . L'application transfère à l'application Historique le record entier modifié. Cette dernière:

-repère les zones modifiées qui sont à historiser
-vérifie que l'historique crée pour chaque zone sera le plus récent pour chacune d'elles (prise en charge des séquences). Si ce n'est pas le cas, il s'agit par conséquent d'une modification **dans** l'historique et donc il n'accepte pas car pour cela il faut utiliser la fonction "correction de l'historique".

Situation après cette modification:




*Fichier concerné*

Par conséquent la dernière situation dans l'historique équivaut à la dernière situation des champs historisables de chaque fichier.

Inconvénient : redondance.
Avantage: sécurité sur ces champs historisés.

2) Corrections de l'historique.

C'est une opération fort délicate et très surveillée car, par cette fonction on peut modifier l'historique (en excluant la situation finale qui était traitée par la fonction précédente).

Cette fonction n'est accessible qu'a un nombre assez restreint de personnes du service compétent.

a) Insertions et MAJ.

L'utilisateur précise la date à laquelle l'opération doit être effectuée:

-si la date est inférieure à D_sf
alors OK
sinon ERREUR (on retombe dans le cas 1) où on veut
modifier Rec_sf)

- Si la date est égale à la date existante dans l'historique
alors c'est une Maj
sinon c'est une insertion.

-repérer les zones modifiées
-effectuer ce contrôle de cohérence

**b) Suppressions:**

L'utilisateur précise la date de la suppression, de même que les zones dont l'historique doit être supprimé. L'application contrôle pour ces zones qu'il existe une date d'historisation égale à celle précisée par l'utilisateur. Cette option "correction de l'historique" a son domaine d'activité inférieure ou égale à D sf-1.

Remarque.

A chaque correction de l'historique le fichier garde la cohérence dans le sens où la valeur de la donnée à la date D est différente

- de la valeur de la donnée à la date D+1
- et de la valeur de la donnée à la date D-1.

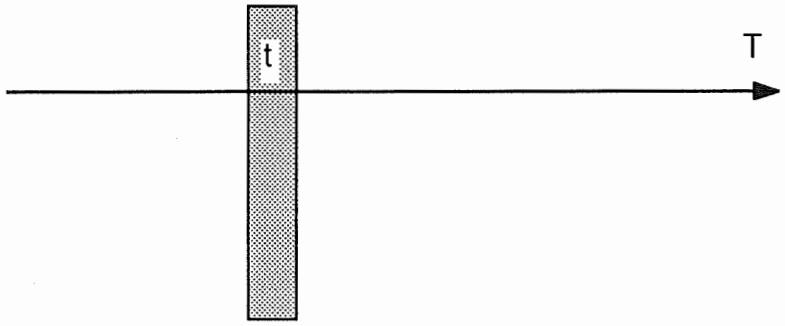
Ainsi on élimine les redondances. C'est le principe du **Redondancy check.**

## 5.2. L'historique du point de vue Application: CHRONOS.

Chronos est l'application se chargeant de la gestion de l'historique dans Arpege. Elle a été écrite avec Natural V2. Cette application permet la création, la maintenance et l'exploitation des enregistrements de l'historique. Elle assure la cohérence entre ces enregistrements ainsi qu'entre les champs qu'ils contiennent. Les objectifs de chronos sont les suivants:

-maintenance et suppressions d'enregistrements de l'historique

-*image construction* : construction d'une image du record (c'est-à-dire des champs historisés + champs non historisés) à une date donnée.




-trace historique: l'évolution des zones suivant un certain critère.

Cette application est composée d'un certain nombre de programmes permettant de couvrir toutes ses fonctionnalités. Ex. *HISTUP* qui s'occupe de la maj de l'historique, *HISTDEL* qui se charge de la suppression dans cet historique. La liste complète de tous les programmes composants de l'application se trouve dans un rapport *"Chronos : A history system prototype using Natural 2. Internal and programmer's guide"* se trouvant dans la documentation Arpege au P.E. Ce document, outre la liste des programmes de l'application historique, se veut être un approfondissement concernant Chronos. **C'est une base pour les concepteurs d'applications souhaitant traiter le problème de l'historique.** On verra dans la partie 4 comment on se sert de Chronos pour historiser au sein d'une fonctionnalité nouvelle.

## 6. L'intégrité de la Base de donnée

Si l'on se penche sur la littérature, on constate que les définitions concernant cette fonction indispensable de tout Sgbd convergent.

En effet, selon [DATE] "*the problem of integrity is the problem of ensuring that the data in database is accurate-that is, the problem of guarding the database against invalid updates. Invalid updates may be cause by errors in data entry, by mistakes on the part of the operator or the application programmer, by system failures, even by deliberate falsification (problem of security)*"

Selon [DELOBEL & ADIBA], ils distinguent plusieurs aspects de l'intégrité d'une BD.

Le premier aspect doit garantir la consistance des informations stockées dans la BD. Par exemple la valeur de l'âge d'une personne doit être compris entre 0 et 120. Dans ce cas on parlera de **contraintes d'intégrité** (CI) ou cohérences auxquelles les données doivent obéir.

Le second aspect consiste en la gestion de la concurrence (**Concurrency**). Si plusieurs utilisateurs travaillent dans un même système au même moment, le sgbd doit assurer que les actions de l'un n'influencent pas ou n'interfèrent pas avec celles d'un autre. C'est le problème de la Concurrency.

Le troisième aspect porte un remède au problème des pannes des ordinateurs. Un ordinateur peut être la cible de difficultés d'ordres techniques ou software. Dans ce cas il est indispensable que le sgbd soit muni de la fonction Récupération (**Recovery procédures**). Cette dernière consistant à remettre la BD dans l'état cohérent précedent la panne. Il existe plusieurs techniques de récupération, les lecteurs intéressés consulteront [DATE] ou [DELOBEL&ADIBA].

Le quatrième et dernier aspect de l'intégrité consiste en la **sécurité**. Les données doivent être protégées contre les personnes non autorisées à les consulter ou modifier.

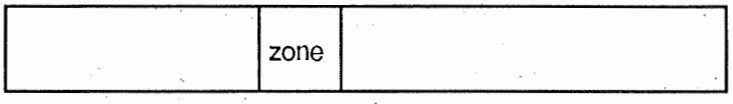
Ces quatre aspects sont généralement présents dans tout sgbd qui se respecte, et ils le sont dans Adabas. Néanmoins le P.E. a développé des fonctions de sécurité (infra 6.2.), de cohérences (infra 6.1.), et de récupération (infra 6.3., le journaling) de manière à traiter ce problème d'intégrité plus efficacement.

## 6.1.Les contraintes d'intégrités ou cohérences au sein d'Arpege.

On peut distinguer trois niveaux de cohérences qui ont étés implementées par des routines de test au sein de l'application Arpege.

1) Niveau 1 (N1) : le champs seulement.

Record



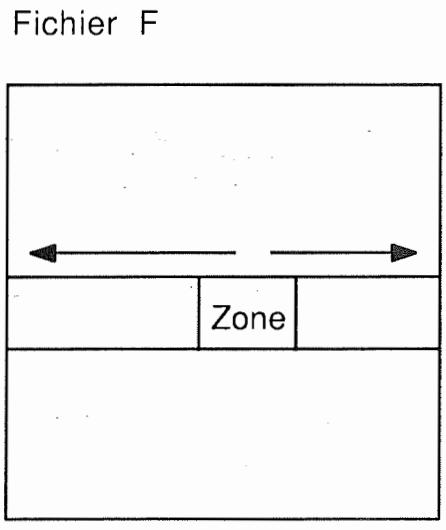

Ce type de tests concernent la plausibilité de la valeur rentrée pour la zone.

Ex. L'âge d'une personne >0 et < 150.

Ce niveau de cohérences est couramment pris en considération au sein des applications.

2) Niveau 2 (N2) : cohérence de la situation au sein d'un fichier.

C'est le test de la valeur de la zone en rapport avec les autres zones du même fichier.



3) Niveau 3 (N3) : cohérence de la situation globale.

Concerne les tests sur la plausibilité de la valeur rentrée pour la zone en se rapportant aux autres champs du record et même en fonctions des autres champs liés à ce record mais étant dans d'autres fichiers. Donc test sur la valeur rentrée pour une zone en fonction:

-des autres champs du même record du fichier
-des autres champs en relation avec le record dans un autre fichier.

On comprendra assez aisément que la difficulté des tests de ce niveau de cohérence s'avère importante. En effet, plus le record possède de champs, plus complexe devient ce test. Cette phase de test doit être fondée sur une réflexion conjointe entre les utilisateurs et les informaticiens. Car principalement, ce sont les utilisateurs, qui en fonction des valeurs possibles d'une zone, connaissent les contraintes de celles-ci en fonction des autres.

Dans l'état actuel des choses, ce type de test est très peu implémenté dans les applications de gestion de personnel existantes au Conseil. Par contre dans Arpege il est pris en considération.

Pour implémenter ces contraintes d'intégrité, il faut effectuer à la saisie des tests de plausibilité et de vraisemblance sur la situation résultante.
A chaque test correspond une routine ne concernant qu'une zone.

On peut distinguer deux types de messages-tests:

-les "*bloquants*" qui interdisent la mise-à-jour du fichier. Tant qu'on n'a pas corrigé, on se trouve bloqué.
-les "*warnings*" qui affichent un message de mise en garde mais qui peuvent être outrepassés.

Les messages bloquants sont utilisés pour les zones où les valeurs sont soigneusement vérifiées et où donc leur cohérence est importante. Les "warnings" sont utilisés pour des valeurs de zones moins importantes et qui n'ont pas de graves conséquences pour la cohérence des données.

Les cohérences de niveau 2 et 3 étant très peu (ou pas) prises en considération dans les applications de gestion de personnel au Conseil, il deviendra indispensable de concevoir des routines de test pour chaque zone. Etant donné que les routines de tests de niveau 2 et 3 existent au P.E. on pourrait le adapter pour les données du Conseil. Les tests effectués au Conseil seront d'ailleurs très semblables à ceux existant sous Arpege au P.E. Par contre les routines de test de niveau 2 et 3 devront être précédées d'une phase d'élaboration de ces équations de cohérences.
Nous aurons l'occasion de reparler ultérieurement du problème des cohérences lorsqu'on abordera le problème "d'initialisation de la B.D" (point 5.2. de la Partie 3)

## 6.2. La sécurité.

### 6.2.1. Le concept de sécurité.

Tous les sgbd (ou dbms) doivent fournir une fonction de sécurité. Cette dernière doit garantir la confidentialité des données. Certains utilisateurs ont droit à accéder à certaines données et pas à d'autres. Cet aspect de sécurité est parfois crucial dans certains domaines comme les banques par exemple. Dans une organisation comme le PE ou le Conseil, il est important d'avoir une gestion de la sécurité fiable car les informations que la BD contient concernent la vie privée des fonctionnaires.

Selon [DATE] et [DELOBEL&ADIBA], il existe différentes techniques permettant de formuler et de renforcer les contraintes de sécurité.

La première, exposée ci-dessous (infra. 6.2.2.), concerne *l'identification et l'authentification*. La seconde (infra 6.2.3.) technique, avancée par [DELOBEL&ADIBA], consiste en le *contrôle de flux de données*. La troisième tente de mettre en évidence la technique des *contrôles d'inférences* (infra 6.2.4.). Et la quatrième, et dernière technique, sera consacrée a *l'encryptage des données* (infra 6.2.5.).

### 6.2.2. L'identification et l'authentification

Avant d'accéder à l'application, l'utilisateur s'identifie, en ce sens qu'il fait savoir au système qui il est. Souvent cela consiste à rentrer son nom ou son numéro de matricule. Dés réalisation de cette opération, l'utilisateur doit prouver qu'il est bien la personne qu'il prétent être. Cela consiste à s'authentifier, en rentrant généralement son *password* (mot de passe). Si la personne ne donne pas le mot de passe correspondant à la

personne identifiée, l'accès n'est pas donné et il convient alors de recommencer (un nombre limité de fois) le processus d'identification/authentification.(1)

Une fois que la personne identifiée s'est aussi bien authentifiée, alors le système associe chaque utilisateur à une classe d'utilisateurs (*user profile*). Tous les utilisateurs appartenant à une même classe ont les mêmes droits d'accès, les mêmes privilèges. Il existe une classification des données. A chaque donnée lui correspond un niveau de sécurité:

*top secret*
*secret*
*confidential*
*unclassified*

On pourrait à ce stade réflechir sur la fiabilité d'un tel processus d'identification/authentification. En effet, on constate que:

**(1)** les gens ont peu d'imagination, ils choisissent souvent un mot de passe reprenant le prénom de leur épouse, ou de leur fils, ...
**(2)** les utilisateurs ne changent pas assez souvent de mot de passe
**(3)** si une personne a envie de connaître le mot de passe d'une autre il suffit de regarder lorsqu'elle rentre son mot de passe à l'écran.
**(4)** il y a toujours moyen d'aller trouver ce mot de passe (souvent crypté) dans un fichier du système.

Pour remédier quelque peu à ces défaillances, et particulièrement à la troisième, [DATE] propose une procédure d'identification/authentification plus fiable que la précedente.
Après l'identification, le système fournit (de manière aléatoire) à l'utilisateur un nombre X. L'utilisateur à ce moment effectue sur ce nombre X une transformation mentale (T) assez simple et rentre au système le résultat Y équivalent à T(X). Le système effectue la même transformation T sur X et vérifie que ce qu'il a trouvé est équivalent à Y. Dans ce cas, la personne s'est bien authentifiée. Avec un tel procédé, quand une personne essaye de s'emparer du mot de passe d'une autre, elle ne voit à l'écran que les nombres X et Y, mais ne connaît pas le lien entre le premier et le second. Sa seule chance de réussite est que lorsqu'elle essaye de s'introduire, le système lui fournisse comme nombre aléatoire le même nombre X de l'autre personne. Mais comme ce nombre est aléatoire...

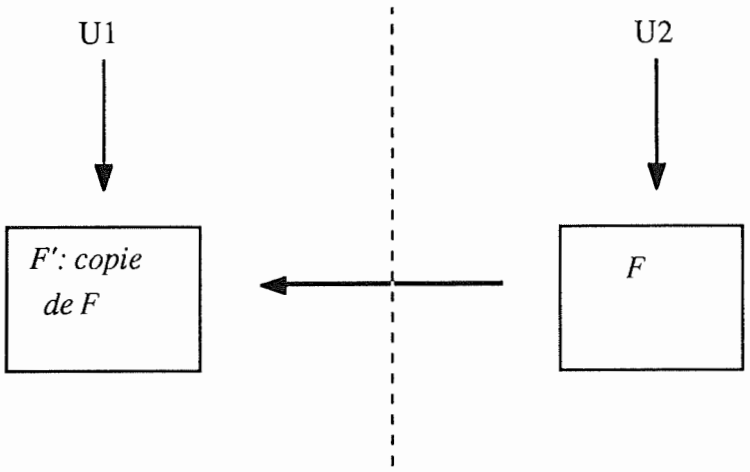
(1) Pour mieux faire, il serait intéressant d'enregistrer les données propres à cet echec (heure,identification, mot de passe). Aisi la personne concernée par cette identification pourra savoir si on a utilisé son identification de manière abusive.

Les recommandations qu'on pourrait émettre pour les trois autres défaillances sont que :

- les utilisateurs choisissent les mots de passe avec un peu plus d'imagination
- ils les changent fréquemment (au moins une fois tous les quinze jours).
- le sgbd ait une bonne technique de cryptage des données.

### 6.2.3. Contrôle des flux de données

Ce problème est illustré par l'exemple suivant: U1 et U2 sont deux utilisateurs du sgbd. U1 n'as pas le droit d'accéder au fichier F, mais U2 a ce droit. U1 afin de pouvoir y accèder demande à U2 de lui faire une copie de ce fichier et de la mettre dans sa "directory". Ainsi U1 pourra accéder aux mêmes informations que celles contenues dans le fichier F.




F : fichier source
F': fichier objet

On peur émettre une règle générale afin de mieux contrôler ce flux de données : *le transfert d'informations d'une 'source' vers un 'objet' n'est possible que si le niveau de sécurité de 'l'objet' est au moins aussi élèvé que celui de la 'source'.*

### 6.2.4. Contrôle d'inférence ou contrôle dans les bases de données statistiques.

Une base de donnée statistique est une BD contenant un certain nombre de records sur lesquels les utilisateurs ne peuvent effectuer que des opérations statistiques (COUNT, SUM, AVERAGE). Le problème est ici qu' un utilisateur, à partir de ces opérations statistiques globales, peut retirer des informations confidentielles.

Par exemple:

*BD statistique concernant une société*

| NOM | Sexe | ... | Profession | Salaire | Audits |
|---|---|---|---|---|---|
| Martin | M | | Economiste | 50 | 1 |
| Norton | F | | Ouvriere | 30 | 0 |
| Pantin | M | | Magasinier | 30 | 0 |
| Dupond | F | | Secrétaire | 30 | 0 |
| Durand | M | | Directeur | 100 | 3 |
| Jones | F | | Avocate | 50 | 1 |

Si l'on veut connaître le salaire de Durand (la consultation de ce champs est interdite) il suffit de faire (en utilisant le langage SQL):

```
Q1:  SELECT COUNT (*)
     FROM BD
     WHERE SEX ='M'
     AND PROFESSION = 'DIRECTEUR'
                                  Réponse: 1

Q2:  SELECT SUM(SALAIRE)
     FROM BD
     WHERE SEX='M'
     AND PROFESSION ='DIRECTEUR'
                                  Réponse: 100
```

On constate donc que même en utilisant des Count, Sum, Average,..., on arrive à retirer des informations confidentielles individuelles. En réalité si l'utilisateur arrive (via la requête Q1) à trouver un prédicat qui identifie uniquement la personne désirée alors, il lui suffira que dans la requête suivante (Q2) il demande son information recherchée dans le SELECT.

On voit donc que si avec la requête Q1 on arrive à identifier le record désiré alors tous les champs confidentiels de ce record ne sont plus en sécurité.

On pourrait, de manière à remédier à ce problème, n'accepter que des requêtes dont le nombre de records sélectionnés est compris entre une borne b et une borne n-b

(où n c'est le nombre total de records du fichier, et avec b>1). Ainsi une requête telle que Q1 serait refusée car réponse<b.

Malheureusement cette restriction est inadéquate. L'utilisateur peut encore obtenir les informations confidentielles concernant une personne.

Supposons b=2, c'est-à-dire que la réponse à une requête ne sera donnée que si le nombre de records sélectionnés est supérieur ou égal à deux, et inférieur ou égal à 4 (6-4).

```
Q3:  SELECT COUNT(*)
     FROM BD
     WHERE SEX='M'
                                        Réponse:4
Q4:  SELECT COUNT(*)
     FROM BD
     WHERE SEX='M'
     AND NOT (PROFESSION='DIRECTEUR')
                                        Réponse: 3
```

Pour ainsi obtenir Q3-Q4=1 en sorte qu'on a identifié la personne desirée (Q3 et Q4 remplacent le rôle de Q1). Maintenant il nous suffit, dans Q3 et Q4, de remplacer les conditions de sélection (dans le SELECT).

```
Q5:  SELECT SUM(SALAIRE)
     FROM BD
     WHERE SEX='M'
                                        Réponse: 180
Q6:  SELECT SUM(SALAIRE)
     FROM BD
     WHERE SEX='M'
     AND NOT (PROFESSION ='DIRECTEUR')
                                        Réponse: 80
```

On effectue la soustraction Q5-Q6 pour avoir le salaire de la personne sélectionnée au moyen de Q3 et Q4.

On peut affiner encore notre dispositif de sécurité (le lecteur intéressé peut consulter [DATE]) afin de rentre la tâche de l'utilisateur plus difficile. Malheureusement ce contrôle ne peut être efficace à 100%, l'utilisateur averti pourra trouver une solution à sa question.

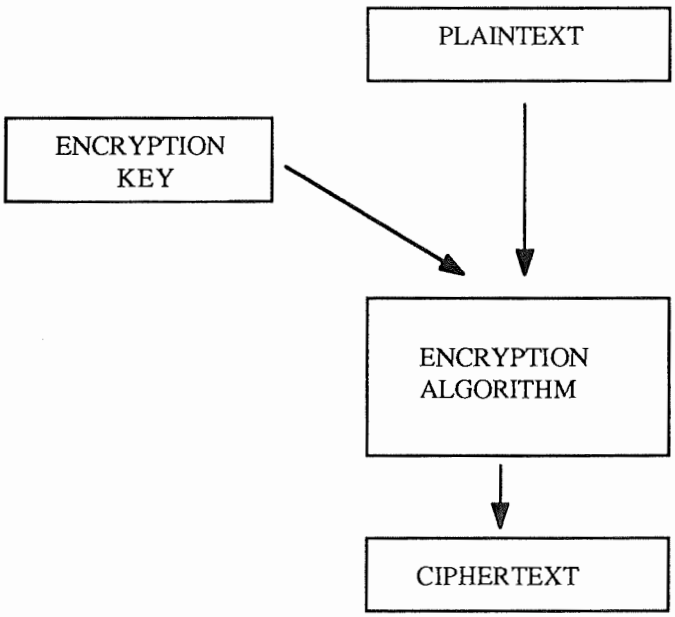
### 6.2.5. L'encryptage des données.

C'est une technique consistant à rendre plus difficile la tâche d'obtention d'informations confidentielles par un utilisateur de "mauvaise foi". Les techniques précedentes tentaient de renforcer les contrôles de sécurité afin de préserver les accès à ces données aux personnes non autorisées. Dans cette partie, on va faire l'hypothèse que l'utilisateur a la possibilité d'obtenir frauduleusement ces données.

L'encryptage consiste à transformer les données de manière telle que si elles tombent entre les mains d'utilisateurs de "mauvaise foi", elles soient inexploitables. Cette technique d'encryptage s'est développée avec la prolifération des réseaux de communications, car dans ce cas les informations sont parfois envoyées via des lignes de communications. Ces lignes, très vulnérables, sont la cible de "tapping" (2) de la part d'utilisateurs. La meilleure technique contre ces procédés est l'encryptage des données.

Introduisons les concepts de cette technique [DATE].

La donnée originale est appelée *plaintext*. Cette *plaintext* est manipulée par l'algorithme d'encryptage (*encryption algorithm*). Ce dernier, en effectuant une transformation en fonction de la clé d'encryptage (*encryption key*) donne comme output la donnée cryptée, le *cyphertext*.




(2) ecoute, recopiage de données de manière fraudouleuse.

C'est donc le *cyphertext* qui sera envoyé dans la ligne de communication. Le destinataire, où l'utilisateur 'legal' de la donnée, possédant la clé d'encryptage et l'algorithme de décryptage, peut retrouver la donnée sous sa forme initiale (le *plaintext*).

**Qu'en est-il de la sécurité dans Arpege?**

Au sein d'Arpege il existe deux niveaux de sécurité:

-au niveau des fonctions: c'est-à-dire que l'accès à certaines applications peut être restreint à un certain groupe d'utilisateurs.

-une fois l'autorisation d'utilisation de la fonction il y a sécurité au point de vue des données:

* restriction sur les fichiers
* restrictions sur les champs du fichier

Ceci en fonction du *User-profile* dont fait partie l'utilisateur. Ces restrictions sont d'ordres divers:

-soit autorisation à lire
-soit autorisation à écrire
-soit autorisation à lire et écrire
-soit autorisation à supprimer et lire et écrire

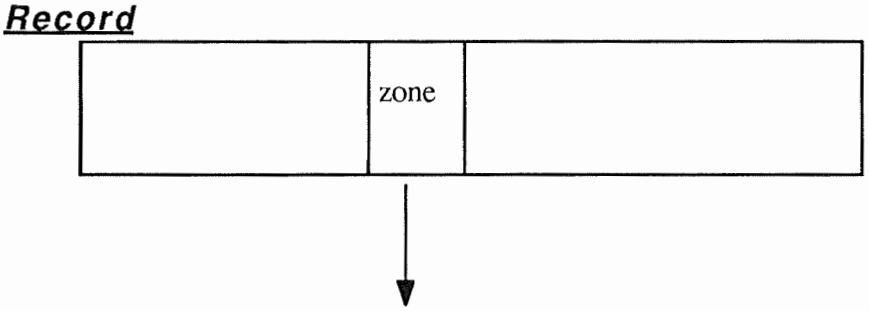
Ce deuxième niveau de sécurité est souvent implementé par des "vues" (sur certains champs) en fonction du *user-profile*. Ainsi l'utilisateur ne sait pas à quels données il n'a pas accès.

Ces deux niveaux de sécurité sont pris en charge dans Arpege par:

1) Natural: le langage de programmation donne la possibilité de protéger (via le "Natural security" qui est un module à option de l'environnement Adabas/Natural) les fonctions développées. Ainsi on peut spécifier l'accès à tel module pour tel ou tel *user-profile* et pas pour tel autre.

2) Giant : tout utilisateur est défini dans un *user-profile*. Chaque *user-profile* hérite de droits d'accès concernant des programmes et/ou des données. La restriction au niveau des fonctions (ou programme) est implementée par la notion du "*Entry-point*" qui ne donne l'accès qu'a une certaine branche de l'arbre qui représente toute l'application Arpege.

3) Adabas "security-by-value": permet de restreindre l'accès à certains records en fonction de la valeur d'une certaine zone du même record et en fonction du *user-profile* de l'utilisateur. On peut donner, par exemple, le droit d'accès à tel groupe d'utilisateurs exclusivement aux records du fichier dont la valeur de la zone xxx est égale à vvv.




Si valeur zone = 'V1' alors record
accèssible aux users_profiles UP1, UP2,...
= 'V2' alors aux users_profiles UP1
= ---

Avec Users_profiles :
UP1=(pascal, jean, ...)
UP2= (Paul)

etc...

La sécurité dans Arpege se situe donc à quatre niveaux:

-niveau des fonctions

-niveau des fichiers

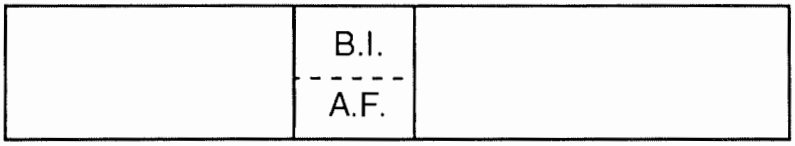
-niveau des zones à l'intérieur du fichier

-niveau des "users profiles"

## 6.3. Le journaling.

Le software Adabas offre la gestion d'un journal où figurent toutes les modification effectuées au cours de la journée. Il se trouve que ce journal a un format qui lui est propre. Pour améliorer cette gestion, le P.E. a développé un programme "sur-mesure" qui s'occupe de la gestion de toutes les modifications. En effet, à chaque modification, ajout, suppression, on tient à jour un fichier ("*le journal*").

Fichier "Journal"




Record

***B.I. : Before Image***
***A.I. : After Image***

Chaque record de ce fichier contient la vue avant (BI) et la vue après la modification (AI).
Ce "journal" est utilisé pour diverses fonctions:

-éditer les "fiches individuelles" concernant le fonctionnaire qui vient d'être la cible de modifications concernant ses données. Ainsi, à chaque modification, la personne recoit une fiche contenant la donnée avec la nouvelle valeur. Ainsi cela officialise cette modification et le cas échéant d'erreur, il peut la signaler.

-il contribue à effectuer certaines statistiques
-il sert pour les transferts vers le système Sysper de la Commission.
-sans oublier la possibilité de récupération ("recovery") des données en cas de crash.

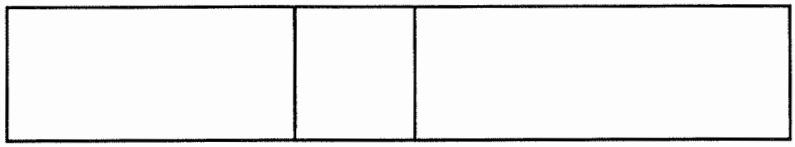
Ce fichier "journal" est quotidiennement, en fin d'exploitation, copié dans un autre journal "mensuel". Sur ce dernier en fin de mois on retire des informations servant à alimenter le "Journal Officiel des Communautés Européennes".

## 7. La gestion des postes. Le "Ring-Book"

### 7.1. La gestion des postes sous Arpege.

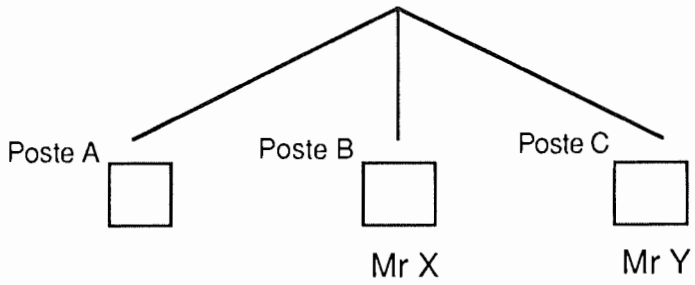
Chaque poste au P.E. est repris dans le fichier des emplois:

Fichier "Emplois"



1Record= 1 poste

Ce fichier comprend même des postes nouvellement crées mais qui ne sont pas encore occupés par une personne. Donc un poste peut exister indépendamment de la personne qui l'occupe.




Dans ce cas, le poste A est caractérisé par l'état d'avancement de la procédure d'affectation le concernant (on a déjà émis un avis, existe-il une personne qui est la bénéficiaire "probable" de ce poste? etc...).

La gestion des postes offerte par Arpege offre la possibilité de:

-créer un poste
-effacer un poste
-modifier un poste
-afficher les caractéristiques d'un poste

Pour chacune de ces fonctions, l'utilisateur interagit via 4 écrans (maps) qui donnent accès aux informations concernant le poste et concernant la procèdure d'avancement de son affectation (si le poste n'a pas encore été attribué).

a) Ecran 1: Organigramme.

-Le premier volet concerne les informations sur le poste. Il sera toujours le même pour les autres écrans.

Occdif: indique si le poste est sous-occupé c'est-à-dire par une personne ayant un grade inférieur à celui demandé par ce poste). Remarque: on ne peut sur-occuper.

-Le second volet : informations provenant du "fichier personnel" concerne la personne occupant ce poste.

-Le troisième volet: utilisé lors de modifications.

b) Ecran 2: Recrutement.

Le second volet: informations concernant l'état d'avancement de la procèdure d'affectation par ordre de priorités:

Les avis (pour les personnes internes): parmi les fonctionnaires il y en aura qui vont postuler. Un programme batch appélé "candidatures internes" va examiner toutes ces candidatures et en fonction de différents paramètres ( leur catégorie, grade, échelon, ...) va émettre la liste de ceux qui sont "recevables". Cette liste ira chez la personne qui a la responsabilité de choisir parmi tous ces candidats (en tenant compte des cotations) la personne qui occupera le poste en question. Dés que le choix a été fait, le nom de la personne apparaîtra dans le second volet.

c) Ecran 3: Suite recrutement.

Si la procèdure d'avis n'a pas donné de candidat, alors on peut enclencher la procèdure "transfert" qui concerne toujours des fonctionnaires internes.
Si on n'a toujours pas de résultats on passe à la procèdure "Recrutement externe" qui ira, soit chercher une personne d'une liste de réserve provenant d'un ancien concours, soit organiser un nouveau concours.

d) Ecran 4: Job description.

C'est une description du travail à effectuer par la personne occupant ce poste.
Remarque: la gestion de l'historique concerne aussi le "fichier Emplois". En effet, sur plus ou moins 80 champs que compte ce fichier, environ une trentaine d'entre-eux sont historisés.

### 7.2. Le Ring-Book.

Il fait partie des programmes "batch" concernant la gestion des postes. Le ring-book consiste en une énorme liste comprenant, dans un certain ordre, tous les postes avec pour chacun d'entre eux, des renseignements sur le poste et sur la personne qui l'occupe.

Dans la pratique, c'est un programme qui imprime tout le "fichier emplois" en fonction du code-service et, pour chaque poste, il ajoute des informations (provenant du "fichier personnel") concernant la personne qui l'occupe.

Cette liste **est l'organigramme du P.E.**

## 8. Les programmes batch.

Ils sont repris dans Arpege sous la rubrique "programmes-ad-hoc"

### 8.1. Introduction.

Au sein d'Arpege, on peut directement donner l'exécution d'un programme batch concernant soit le fichier personnel, soit le fichier emplois. En ce qui concerne le fichier famille, il n'en existe pas ( toutefois, on peut dans le futur, en créer un et cela sans poser trop de problèmes pour la coordination générale).
Ces programmes batch peuvent être assimilés aux programmes de la première couche existants au Conseil. Néanmoins, ils ne les recouvrent pas tous et cela pour deux raisons. Tout d'abord, le Conseil doit disposer de certaines fonctions qui sont unutiles pour le P.E., cela en fonction des besoins des services dépendant du service Informatique. Ensuite, parce que le P.E. a fait la distinction entre des programmes utilisés par un grand nombre de services et les programmes propres à un service particulier (en fonction des besoins de ce dernier).

Leur ligne de conduite est de ne mettre dans la partie des programmes batch uniquement que les programmes communs, publics, et ceux qui sont couramment utilisés. Les autres programmes (plus personnalisés à un service), sont à charge du service qui en est demandeur. De tels programmes se résument souvent à l'édition d'une liste, dés lors leur réalisation avec S.N. ne devrait pas être très complexe. Dans le cas contraire où la réalisation du programme s'avère difficile on fait appel au service Informatique qui se charge de cette réalisation (avec S.N si possible, sinon avec Natural V2)

C'est ainsi qu'au P.E. les services demandeurs d'informations (listes...) ont leur propre terminal qui leur permet (dans le domaine de sécurité qui leur est fixé), au sein de leur "directory", de créer les programmes en S.N. propres à leurs besoins journaliers. Chaque service a son répertoire de "programmes privés" lesquels ne sont pas accèssible aux autres services. Souvent lors de la création d'un nouveau programme, on va voir dans le répertoire s'il existe un programme qui lui ressemble, on fait une copie et on l'adapte. C'est une procèdure assez rapide.

A ce stade, une réflexion doit être effectuée au sein du service Informatique du Conseil. Peut-il, oui ou non, adopter la même ligne de conduite que celle du P.E.? Ceci n'est pas une mince question, car elle relève de problèmes organisationnels. En effet, si le service Informatique décide de "décentraliser" le système d'information de manière à ce que les services puissent eux-mêmes effectuer certaines listes, il n'est pas exclu que les services visés affichent une certaine réticence (car ils devront se former à S.N. ...). Néanmoins, si une telle ligne de conduite était adoptée, le service Informatique se verrait déchargé de problèmes spécifiques (il peurra se consacrer, dès lors, entièrement à des problèmes d'ordre général) et les services acquèreront une certaine indépendance pour trouver la solution à leurs problèmes.

Supposons, et **probablement tel sera le cas**, que cette ligne de conduite soit envisagée par le Conseil. Les services, en fonction de leur connaissances de S.N., vont trouver tel ou tel programme "complexe" à réaliser et de ce fait vont s'en décharger aux dépens du service Informatique. Ce phénomène dépend de divers facteurs (niveau de connaissance de S.N. aptitude à la logique de programmation, ...). Il sera donc necessaire de trouver un équilibre entre les programmes réalisables par les utilisateurs et ceux qui doivent être pris en charge par le service Informatique.

Remarque: Il sera question, au cours de la Partie 4, d'intégrer une nouvelle fonctionnalité au sein de ces programmes ad-hoc.

## 8.2. Les programmes.

a) Fiches individuelles.

Elles sont émises afin de signifier officiellement aux fonctionnaires les changements survenus dans les informations les concernant. Ces fiches proviennent du "journal" qui rassemble tous les changements effectués dans la B.D. Arpege (cfr. point 6.3.). Par cette fiche, les fonctionnaires suivent l'évolution des informations les concernant et de ce fait peuvent intervenir s'ils constatent des erreurs.

b) Rapports de notation à la demande.

Cela consiste en une extraction d'informations provenant de la B.D. d'Arpege qui va compléter un formulaire préétabli à l'avance. C'est une feuille qui sert de base à la procédure des notations. Ce rapport ira directement chez le supérieur hiérarchique à des fins de promotions, mutations, etc..

c) Rapport de notation mensuel.

C'est la même procédure que la précédente, à la différence que celle-ci est automatique et qui s'applique donc sequentiellement pour un certain nombre de fonctionnaires.

d) Tableau des promotions.

C'est un algorithme assez complexe qui calcule les gens que l'on peut promouvoir (en tenant compte, bien sûr, des C.C.P.).

e) Bulletin mensuel.

Il se base sur le "Journal mensuel" pour en extraire une série de modifications survenues dans la B.D. d'Arpege (promotions, mutations, retraites ...) et qui iront alimenter le journal officiel "Bulletin mensuel du personnel des Communautés Européennes". Cette procédure est aussi effectuée au Conseil, et ce programme batch est directement utilisable par le Conseil sans trop d'adaptations.

f) Tableau grade-nationnalités.

Donne un tableau reprenant par nationalité le nombre de fonctionnaires par catégorie, grade...

g) Ring-Book.

Déjà abordé précédemment (voir supra point 7.2.)

h) Candidatures internes.

Déjà abordé précédemment (voir supra point 7.1.b. et 7.1.c. ).

i) Transferts vers Sysper.

Sysper est un système informatique de la Commission consacré à la gestion et à l'administration du personnel et des Emplois des Communautés Européennes.
Ce système utilise une base de données gérée sous Adabas. Dans Sysper sont gérés les fonctionnaires et agents (temporaires, auxiliaires, locaux) de la Commission et des autres institutions (P.E., Conseil, ...).

A cette fin, le P.E. et Conseil doivent fournir à Sysper une extraction des données de la B.D.

j) Gestion du Journal Arpege.

En fin de journée, on recopie le "journal quotidien" dans le "journal mensuel". On remet ensuite le "journal quotidien" vide et ainsi il est prêt pour le lendemain.

l) Sauvegarde/Restore du journal Arpege.

Tous les mois on effectue le backup de ce "journal Mensuel" à des fins de sécurité.

Remarque:
Certains programmes batch peuvent donner des résultats qui ne sont pas directement des listes. En effet, il existe des programmes batch qui ne font que créer un fichier, ce dernier étant ensuite utilisé par un autre programme ou des applications externes à Arpege. C'est le cas, par exemple, lorsqu'il s'agit d'avoir la liste des promouvables: il y a d'abord une extraction de données (via un utilitaire *Crosstalk*), pour constituer un fichier, qui ensuite sera traité par DBase 3.

## 9. Phase II d'Arpege.

Il est intéressant, avant l'installation de ce système, d'en connaître l'évolution future.

Les objectifs qui avaient été fixés par le P.E. au début du projet Arpege étaient:

-dans un premier temps, permettre la réalisation d'un nouveau système qui correspondrait à la conversion du système ancien (c'est-à-dire, conversion des applications RAPE, PACH, SPOT, JOB). C'est l'actuel système Arpege, a savoir, Arpege I. qui contient plus ou moins les mêmes fonctionnalités que l'ancien système. Sa conception a tenu compte du facteur évolutif du système.

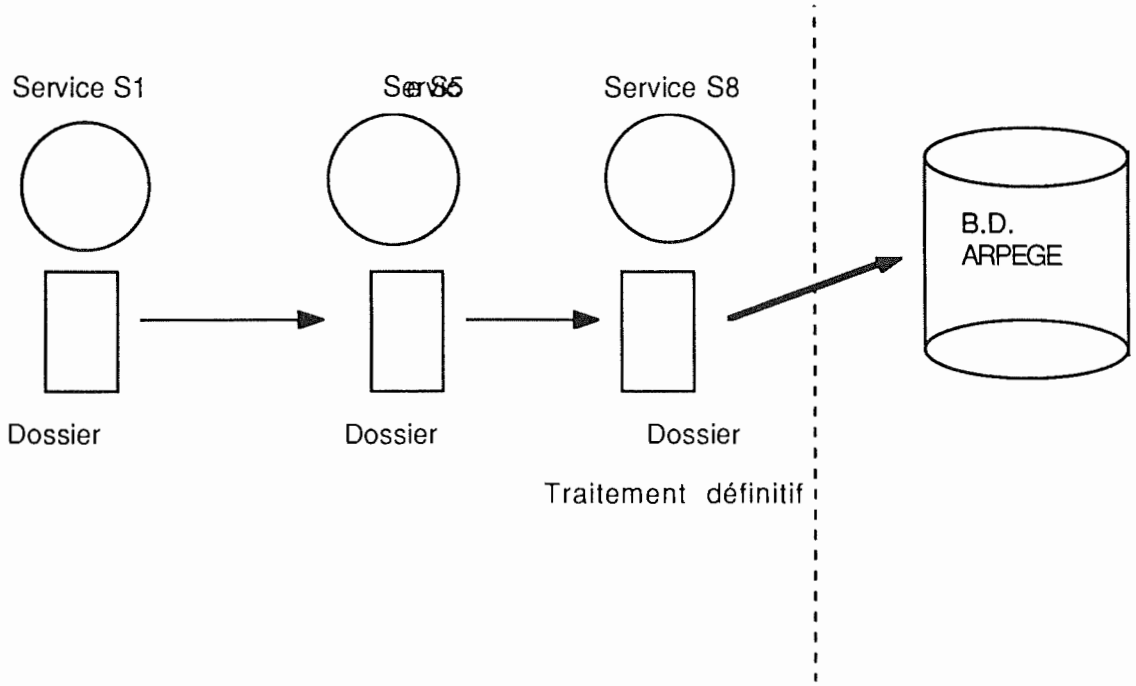
-dans un second temps, permettre la réalisation et l'intégration de nouvelles fonctionnalités attendues par les utilisateurs (Arpege II).

On constate donc que le système actuel (Arpege I) n'est pas le système définitif que le P.E. aimerait avoir. Il n'est qu'un produit intermédiaire.

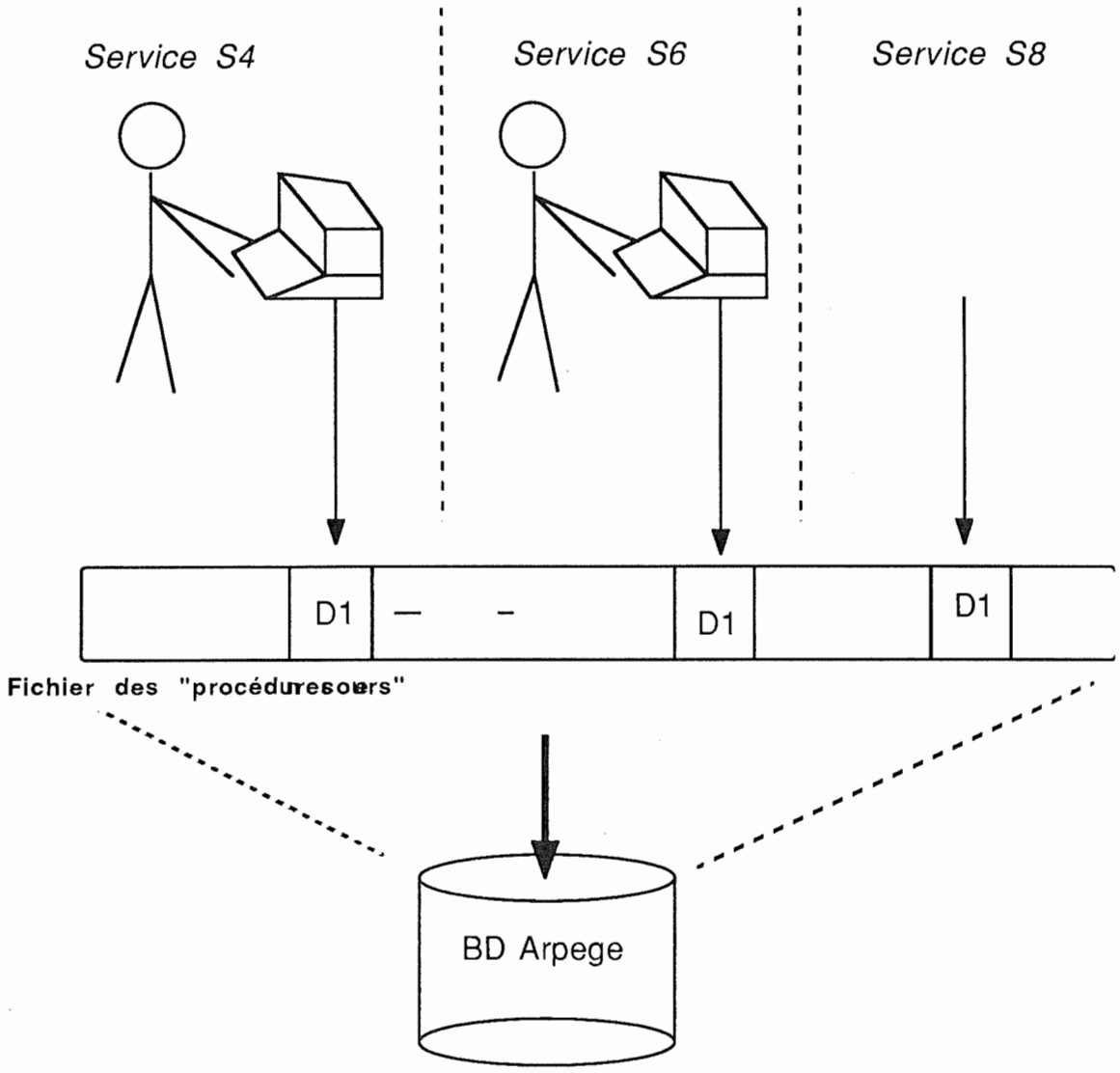
Sa raison d'être est qu'il sert de transition et qu'il est l'image de l'ancien système (en ce sens, il contient plus ou moins les mêmes fonctionnalités). La seule différence avec le système précédent, est que les données ont étés organisées de manière à être gérées par le S.G.B.D. Adabas et que les applications ont étés converties en Natural V2 de manière à interagir avec le SGBD de manière optimale. Bien sûr, lors de sa conception, on a tenu compte du caractère évolutif du système (Arpege II) et on en a profité pour mettre l'accent sur le facteur ergonomique.

Arpege II aura comme objectifs de compléter Arpege I par toutes les fonctionnalités que les utilisateurs ont demandé depuis le début du projet Arpege (et même avant), c'est-à-dire:

-**la décentralisation de la saisie**: actuellement la procédure d'avancement d'un projet est caractérisée par une circulation de papiers.
Dés qu'un service vient de finir son traitement il l'expédie au service suivant qui doit intervenir à son tour dans ce processus:

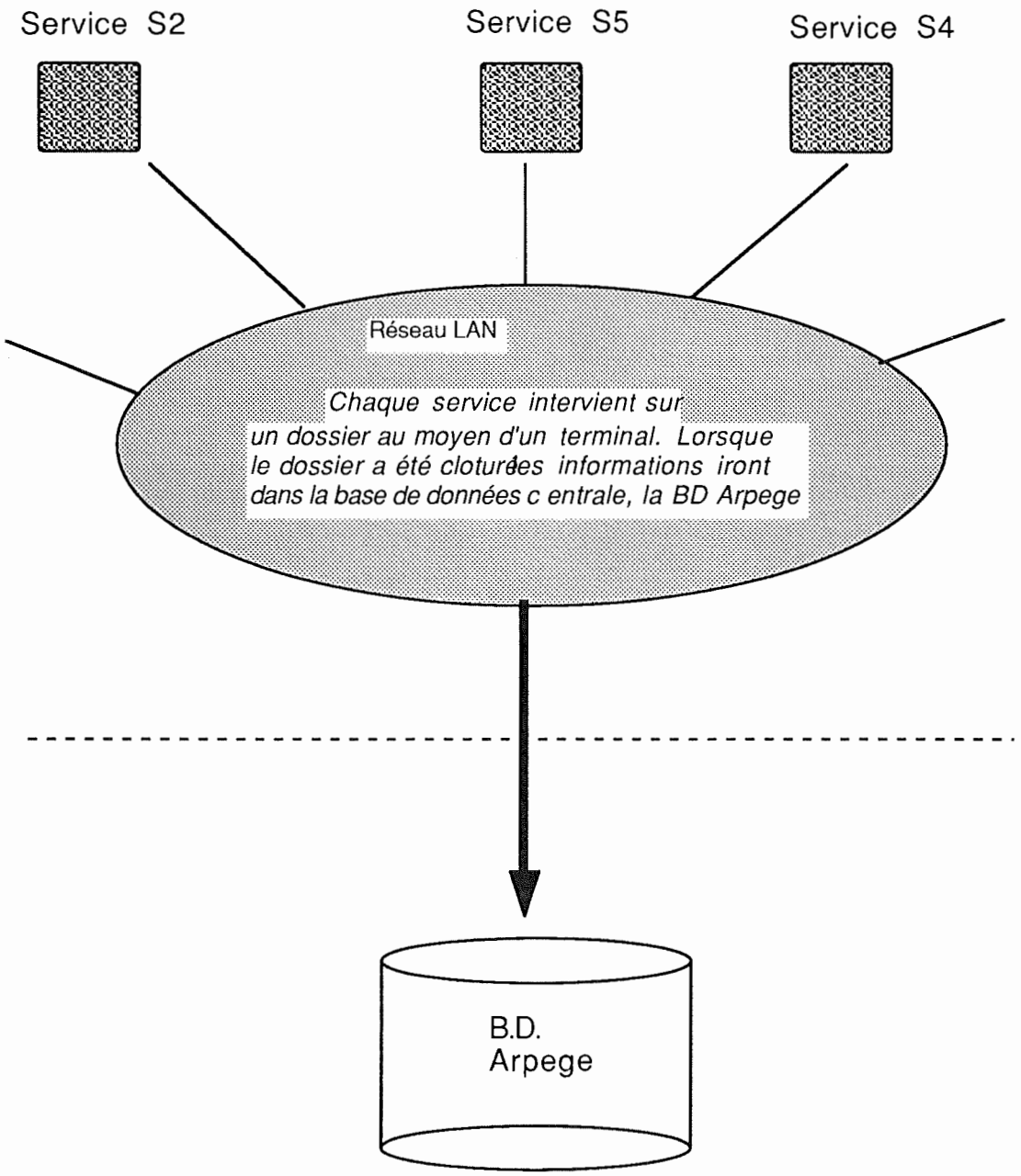


La procédure avec Arpege II sera la suivante:




Lorsque la procédure administrative a abouti, alors, on peut introduire les informations dans la B.D. d'Arpege (ex: mutations, ...).
Cet objectif de décentralisation consiste à suivre l'état d'avancement de la procédure en automatisant le circuit.

Chaque service, interviendra, via le terminal, sur les dossiers qui sont gérés par Arpege. Dés qu'il aura fini son traitement, le dossier ira au service suivant qui devra le traiter. Et à la fin de la procédure, automatiquement (lorsqu'une condition est remplie), Arpege ira intégrer les données dans la B.D. d'Arpege.




Avec une telle gestion on peut suivre l'état d'avancement de la procédure à tout moment.

-**rendre les accès aux données plus ponctuels**: actuellement pour une Maj d'une zone d'un record (par exemple état civil) le système propose à l'utilisateur tout le record (environ 250 champs pour le fichier personnel). L'utilisateur peut alors modifier sa zone au moyen des 12 "maps". On constate donc que l'utilisateur n'a besoin que d'une zone et on lui en fournit 250. Dans Arpege II, on ne présentera à l'utilisateur que les zones nécessaires à sa tâche. Donc les accès seront beaucoup plus ponctuels.

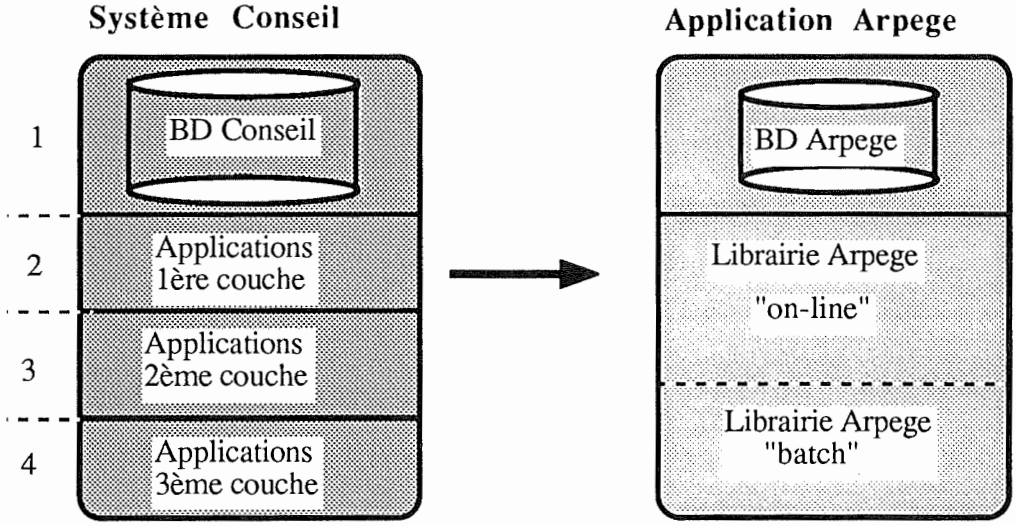
# PARTIE 3: MIGRATION DE LA CONFIGURATION ACTUELLE DU CONSEIL VERS ARPEGE

## 1. Introduction.

Le passage d'un système Informatique à un autre doit être analysé sous deux angles:

-sur le plan de la structuration **des données** (niveau 1 dans le graphique qui suit).
-sur le plan de la structuration **des traitements** ou applications (niveaux 2, 3, et 4 dans le graphique).

On peut structurer dès lors la transition entre le S.I. du Conseil et Arpege de la manière suivante:




Comme déjà vu, les applications du Conseil sont regroupées en trois couches (1ère couche, 2ème couche et troisième couche).

On va décomposer cette transition d'un système à un autre en analysant le passage d'un niveau d'un système à son niveau équivalent dans l'autre système. Donc, on analysera d'abord la transition concernant le niveau 1 (les données), ensuite la transition concernant le niveau 2 (fonctions 1ère couche), etc...

## 2. Adaptation sur le plan des données (niveau 1).

Dans un premier temps, il s'agira ici de comparer les deux structures de données des deux systèmes. En ce qui concerne le Signaletic une liste de toutes ses zones est reprise en Annexes... Pour Arpege, la structure des données est contenue dans la "*Nomenclature Arpege*" ( Voir à ce titre Annexes 2). Ce document reprend par fichier (Personnel, Famille, Emplois) les zones qu'il contient, avec pour chacune d'entre elles, ses caractéristiques (longueur, différentes valeurs possibles, ...). Cette "nomenclature **reflète l'image de la BD Arpege** et de ce fait doit être accessible à un grand nombre de personnes et particulièrement aux utilisateurs. Elle constitue la "Bible"(1) de tout utilisateur et même de l'informaticien travaillant sous Arpege. C'est pourquoi le document est soigneusement tenu à jour.

Cette étape se veut être une étape cruciale, car elle constitue une "charte" concernant les données à manipuler dans le système d'information futur. C'est pour cela qu'il est nécessaire de l'effectuer avec soin.

On peut la décomposer en deux sous-étapes:

*Etape 1: Enumération, épuration et normalisation des données existantes au service informatique du conseil.*

Cela consistera en une réflexion de fond concernant les données manipulées actuellement. Les énumérer, si cela est nécessaire, en supprimer (parfois des services différents manipulent des données presque identiques), et standardiser les valeurs possibles pour tous les champs. A la fin de cette sous-étape, on doit avoir un document reprenant l'ensemble des champs, avec pour chacun d'entre eux, les attributs qui le concernent (longueur, valeurs possibles, ...).

*Etape 2: On va comparer, critiquer et adapter cette structure de donnée avec celle de la BD Arpege.*

La comparaison des deux structures de données doit déboucher sur l'obtention d'un ensemble de zones:

-qui sont contenues dans le Signaletic mais absentes dans la "nomenclature Arpege".

Dans ce cas il faudra étendre la BD Arpege pour qu'elle contienne ces champs

(1) pour utiliser un langage "officieux" au P.E.

-qui sont absentes dans le Signaletic mais présentes dans Arpege.

-qui sont présentes dans les deux structures de données

Dans ce cas il faudra comparer les différentes valeurs que la zone peut prendre. A titre d'illustration mentionnons qu'il existe au P.E. un champs PACTYP (qui donne la relation entre un fonctionnaire et la personne qu'il a à sa charge) pouvant prendre les valeurs suivantes : ENF (enfant), ASS (assimilé), ECJ (enfant conjoint). Le champs correspondant au Conseil, est Lien-Parenté, pouvant prendre les valeurs: Père, Mère, Beau-père,... On constate par cet exemple que, malgré qu'ils existent, ces champs différent par les valeurs qu'ils peuvent prendre. Il faudra à ce stade normaliser cet aspect des choses en tenant compte que l'application Arpege travaille sur les valeurs du champs au PE et que les fonctions des 2ème et 3ème couches du Conseil sont basées sur les valeurs de la zone au Conseil.

Cette étape modifiera la structure des données dans Arpege de manière à ce qu'elle soit adaptée aux besoins du Conseil. A ce stade (une fois de plus), on doit tenir compte de ce que, plus on s'éloigne de la structure Arpege, plus difficile sera notre intégration future d'Arpege (Arpege II, ...).

Dans la pratique, après avoir comparé hâtivement les deux structures, la plupart des données reprises dans le Signaletic sont généralement contenues dans la BD Arpege. Ceci du fait que les données concernant les fonctionnaires et leur environnement (familial, de travail), gardées au P.E., sont similaires à celle reprises au Conseil.
Toutefois, comme Adabas /Natural permet l'adjonction d'un champs au sein d'un fichier de la BD Arpege, ou même l'adjonction d'un fichier, cette adaptation ne posera pas de grands problèmes techniques.
En d'autres termes, si dans le Signaletic on a certains champs contenant des informations qui ne sont pas prises en considération dans Arpege, on peut modifier facilement la structure des données au sein de la BD Arpege.
De par cette flexibilité de la BD Arpege, l'adaptation au niveau des structures des données devrait pouvoir s'effectuer.

## 3. Adaptation sur le plan des fonctionnalités.

En ce qui concerne les fonctions de la première couche au Conseil, il s'agit de les réécrire en SN ou en Natural. Leur complexité étant relativement faible, leur réalisation devrait s'effectuer aisément. Certaines de ces fonctionnalités existent dans Arpege dans

les programmes ad-hoc d'Arpege (programmes batch). C'est le cas par exemple de "grades-nationalités". Dans de tels cas, on peut adopter les fonctionnalités du P.E.

A propos des applications des seconde et troisième couches, le problème est tout autre. En effet, d'une part, on ne trouve pas d'applications qui leur sont équivalentes au P.E. (par exemple A.M. au P.E. est traitée à la Commission). D'autre part, vu leur taille assez conséquente et leur complexité, une conversion en Natural V2 s'avérerait bien difficile et prendrait énormément de temps. Par conséquent, l'idée serait de garder ces applications telles qu'elles sont tout en leur donnant la possibilité d'interagir avec la BD Arpege(1) . Ceci nous amène à parler des différentes hypothèses de migration vers Arpege.

## 4. Scénarios de migration.

Ces hypothèses ne constituent pas les seules manières de permettre le passage d'un système à un autre. Elles ont simplement le but d'apporter des pistes de réflexion.

### A) Phase préparatoire

Ce sont les étapes à réaliser avant la mise sur pied de la phase d'intégration:

-adaptation de la BD Arpege au Conseil (voir point 2.).
-conversion des applications de la 1ère couche du Conseil en Natural et SN.
-Initialisation de la BD Arpege au Conseil (voir point 5.).

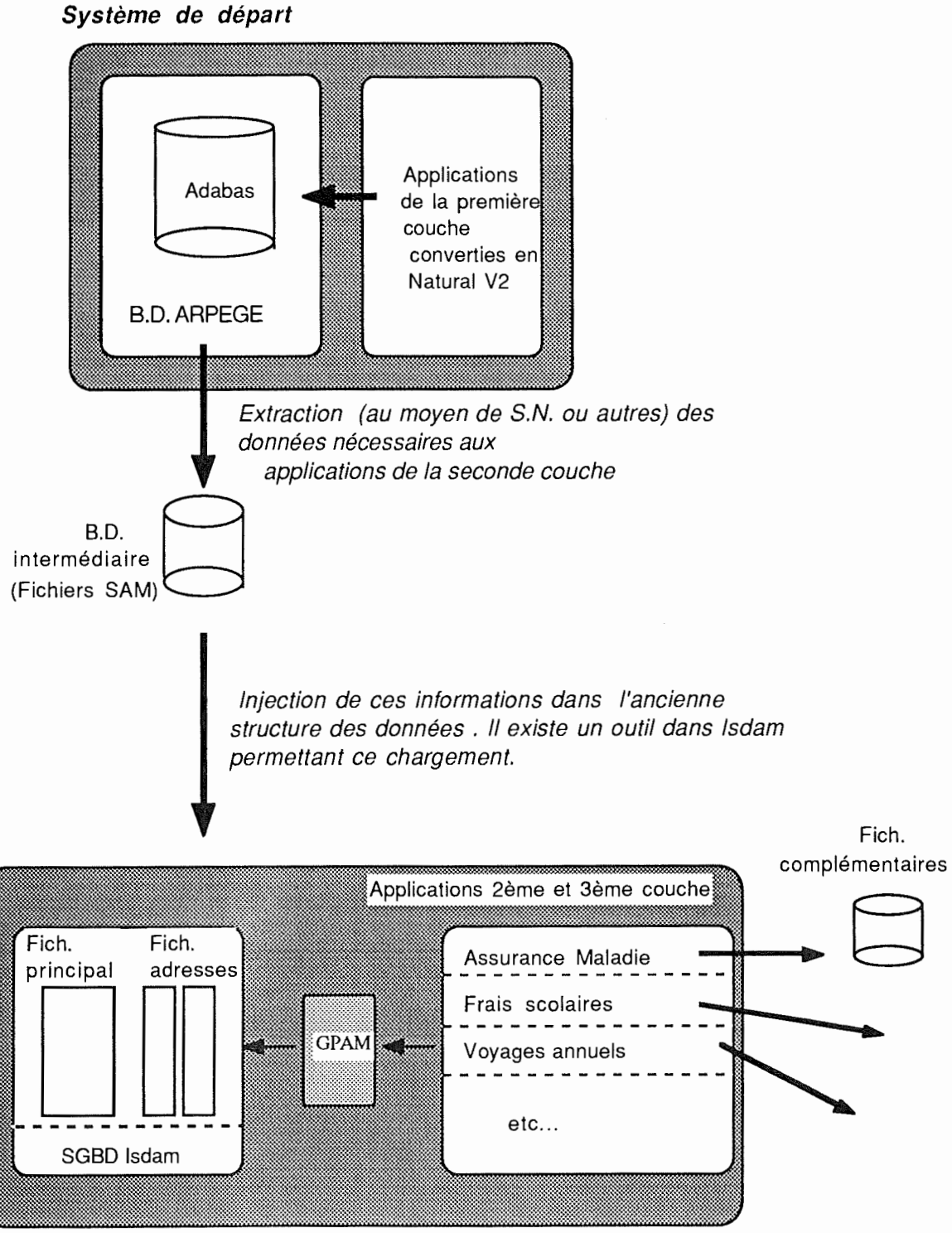
### B) Hypothèse.

*1ère phase: (début 92').*

L'idée est de continuer à utiliser ISDAM en même temps que la BD Arpege. Dans cette dernière, seules les applications de la première couche y travaillent directement. Ceux des seconde et troisième couches consultent toujours le Signaletic (dans lequel, périodiquement, on injecte les données courantes de la BD Arpege). Ceci afin de subvenir aux besoins des applications des seconde et troisième couches en attendant la réalisation

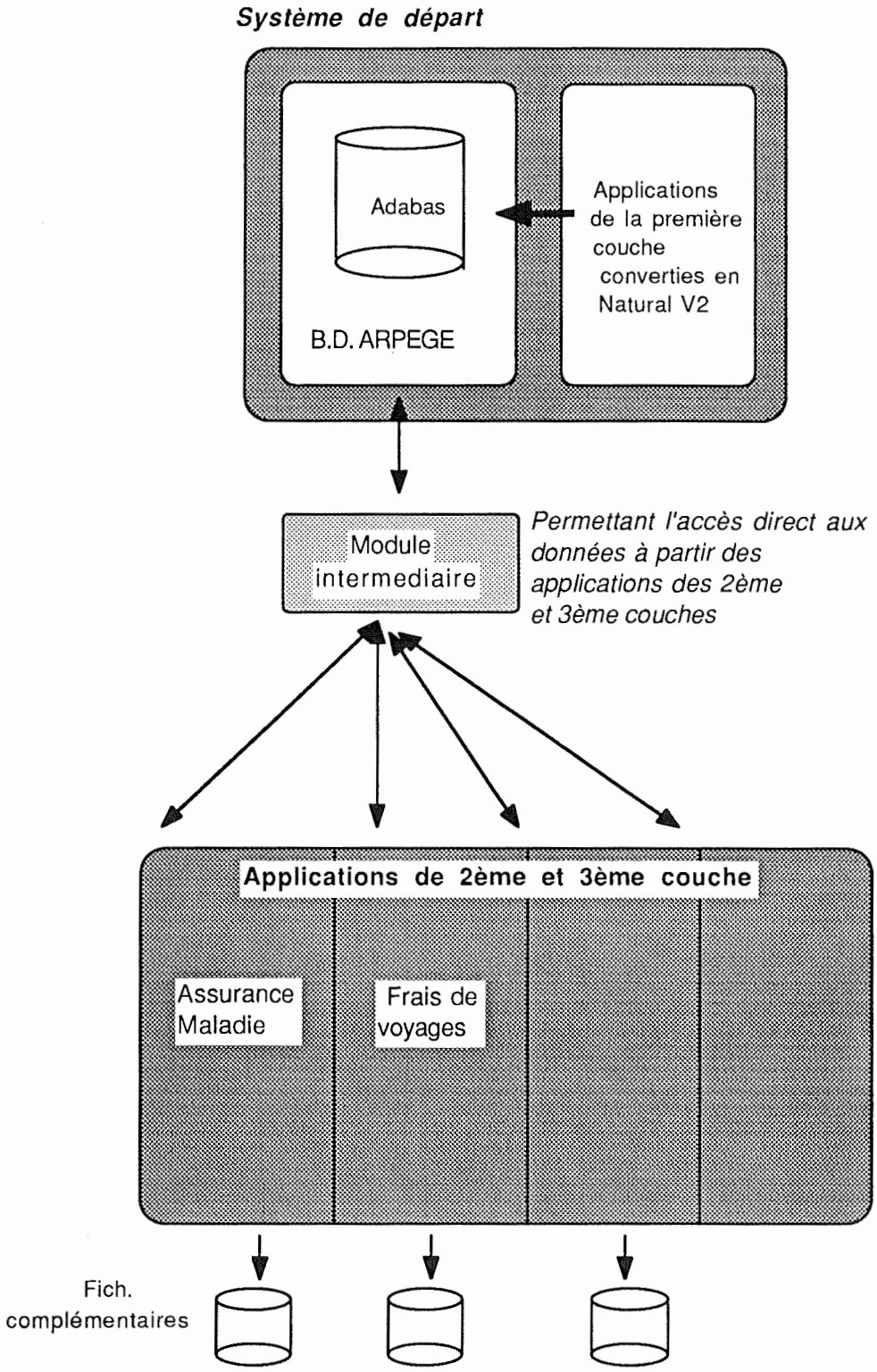
(1) Ne pas oublier le fait qu'a long terme les fichiers complémentaires des applications de la seconde couche devront être intégrés dans la BD Arpege.

d'un "module Intermédiaire" qui remplacerait le module intermédiaire Gpam. Ce dernier permettrait aux applications de la 2ème et 3ème couche d'accéder directement aux informations recherchées dans la BD Arpege.



L'extraction de la BD Arpege n'est pas une opération complexe en raison de l'existence d'utilitaires servant à cet effet (SN, ou un autre produit Adabas/Natural). L'injection de données dans Isdam est permise grâce à l'existence d'un utilitaire (dans l'environnement Isdam) qui effectue cette opération.
Le système pourrait continuer cette exploitation "mixte" jusqu'à la réalisation du "module intermédiaire" permettant aux applications des 2ème et 3ème couches d'accéder directement à la BD Arpege. Dans ce cas on pourra abandonner la structure du Signaletic. Plus vite sera réalisé ce module intermédiaire, plus vite on pourra abandonner l'exploitation mixte Arpege/Signaletic.

Au plus tard en 1993: (car cela devrait être la fin d'utilisation d'Isdam), on ne disposera plus du Signaletic qui sera entièrement remplacé par la BD Arpege. Les consultations de cette BD par les fonctions des 2ème et 3ème couches se feront via le module intermédiaire (qui remplacera GPAM)

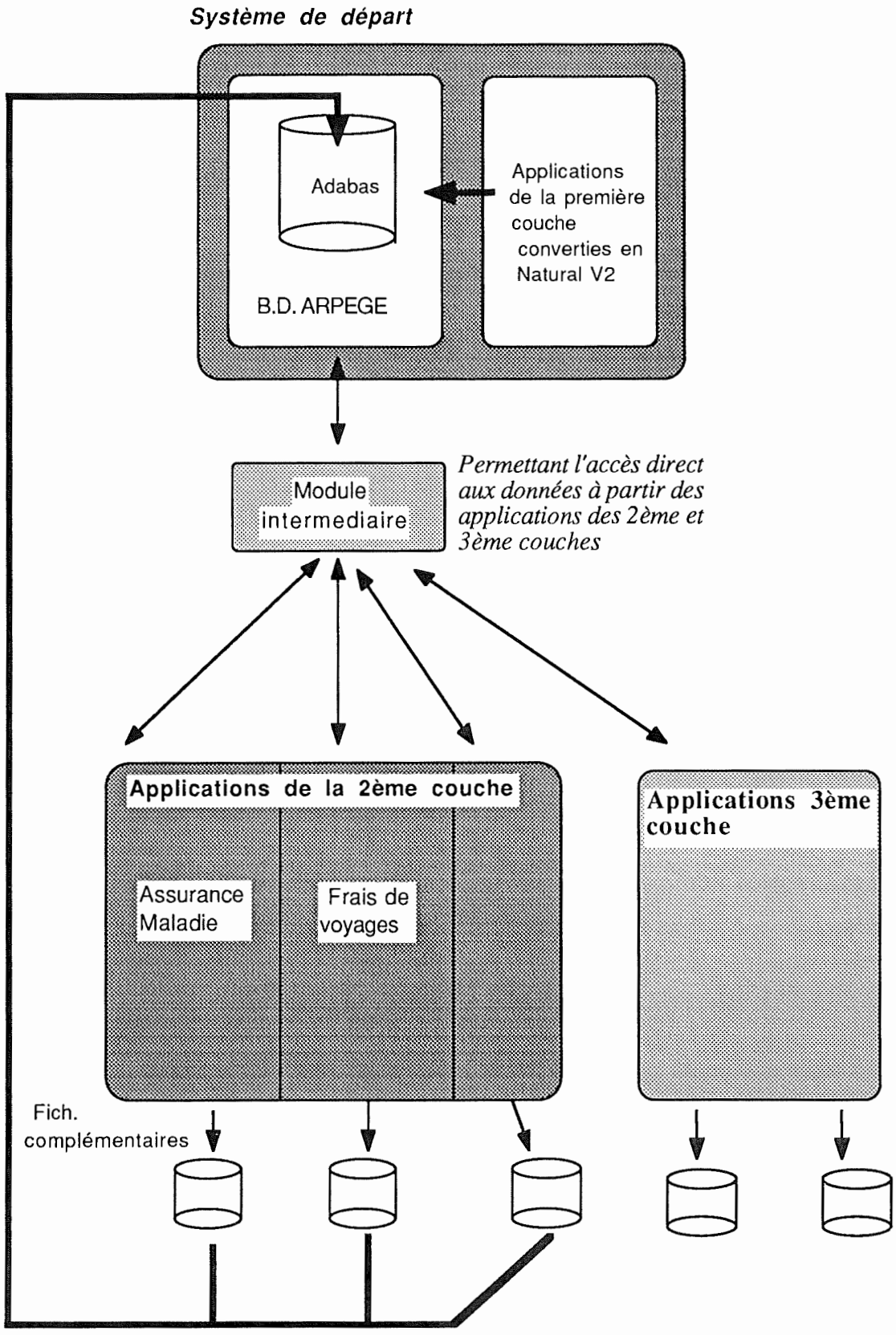



Les applications satellites consultent généralement les mêmes informations (souvent administratives). Ce module intermédiaire contiendrait sommairement l'équivalent de GREAD et GNOM (respectivement, lecture d'un record à partir du numéro de matricule ou du nom).

*2ème phase.*

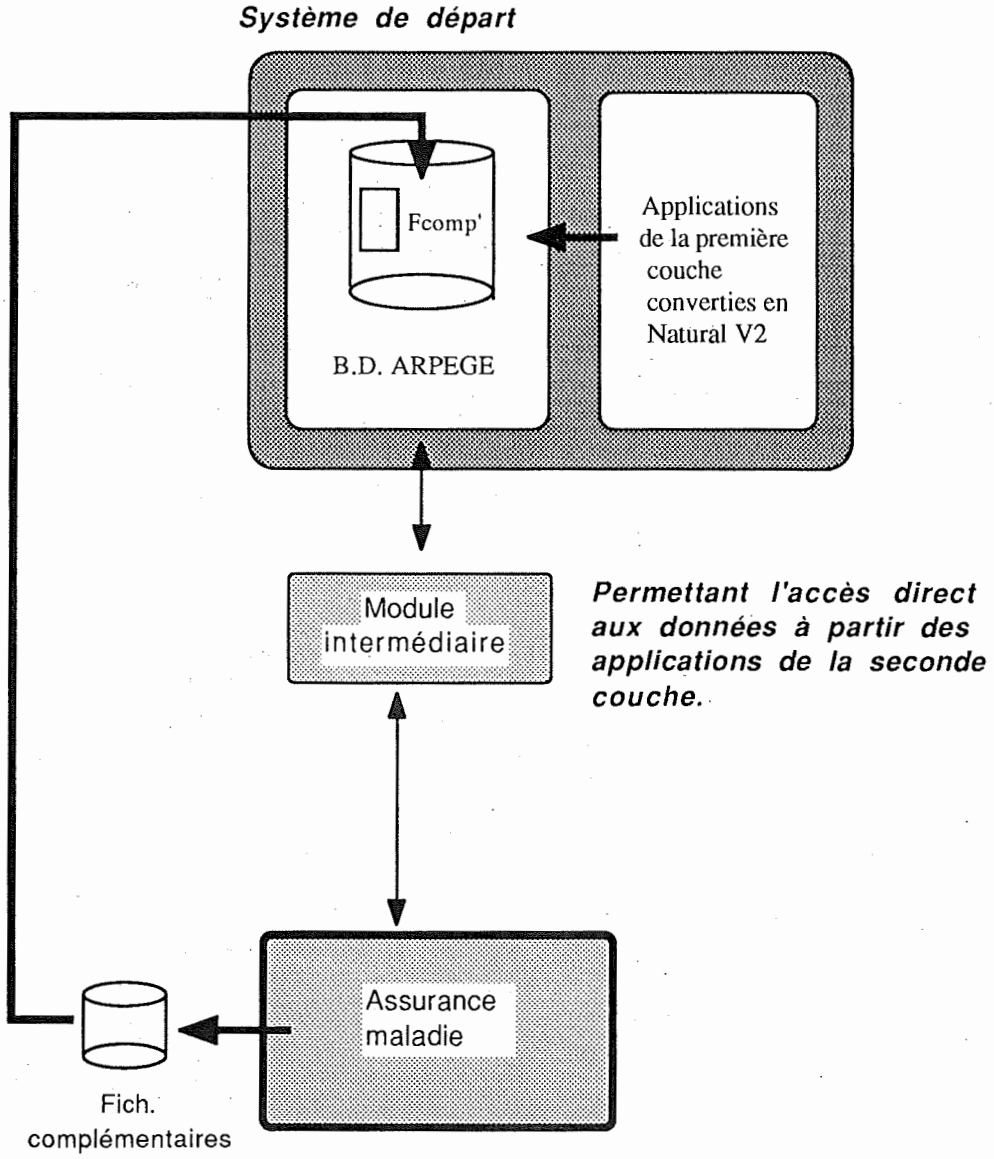
A ce stade, le système peut continuer à être exploité sans contrainte de temps (c'était le cas de la phase précédente à cause de la date limite d'utilisation d'Isdam). Il s'agira dans cette phase d'intégrer progressivement les fichiers complémentaires des applications de la seconde couche.(2)

(2) Il est à noter qu'au sein de la BD Arpege il existe certaines informations qui sont reprises dans les fichiers complémentaires ( c'est le cas des comptes bancaires). Par conséquent, une partie du travail est déjà faite.




***Integration des fichiers complémentaires au sein de la B.D.. Arpege.***

**Pour assurer la prise en compte dans Arpege d'un fichier complémentaire, on pourrait le réaliser de la manière suivante:**




-élargissement de la BD Arpege afin que cette dernière contienne les données (les zones ) du fichier complémentaire (tenir compte de l'historique de ce fichier).

-initialisation de cette nouvelle zone fcomp' de données avec des données réelles contenues dans fcomp.

-réalisation de la fonctionnalité F (c'est un sous-module) au sein du module intermédiaire. Modifications des *appels* dans l'application concernée de la seconde couche.

-par des tests successifs, on peut tester la fiabilité de l'extraction des données contenues dans les nouvelles zones (avec ce sous module).

-lorsque les résultats des tests sont jugés satisfaisants on peut alors

* réinitialiser la zone fcomp' de manière à lui donner les mêmes informations contenues dans fcomp.
* exploiter définitivement le chemin (1), F, (2), fcomp'.
* abandon de l'exploitation du fichier fcomp.

**Quelques considérations concernant le module intermédiaire:**

-au début de cette seconde phase, il ne contenait que les procédures GREAD et GNOM. qui fournissent un enregistrement à la fois dans un buffer de 4400 bytes selon qu'on y accède par Num. de matricule ou par le nom.

-à chaque intégration d'un fichier complémentaire dans Arpege, il sera nécessaire de construire un sous-module de manière à accéder (en consultation, mais aussi en modification) directement aux données utiles dans Arpege. Donc, à chaque prise en considération d'un fichier complémentaire dans Arpege, ce module intérmediaire comprendra un sous-module supplémentaire, propre à l'accès des données via ce dernier. Il sera aussi question de modifier les "appels" (concernant les accès à l'ancien fich. comp.) de manière à effectuer l'appel via ce sous-module.

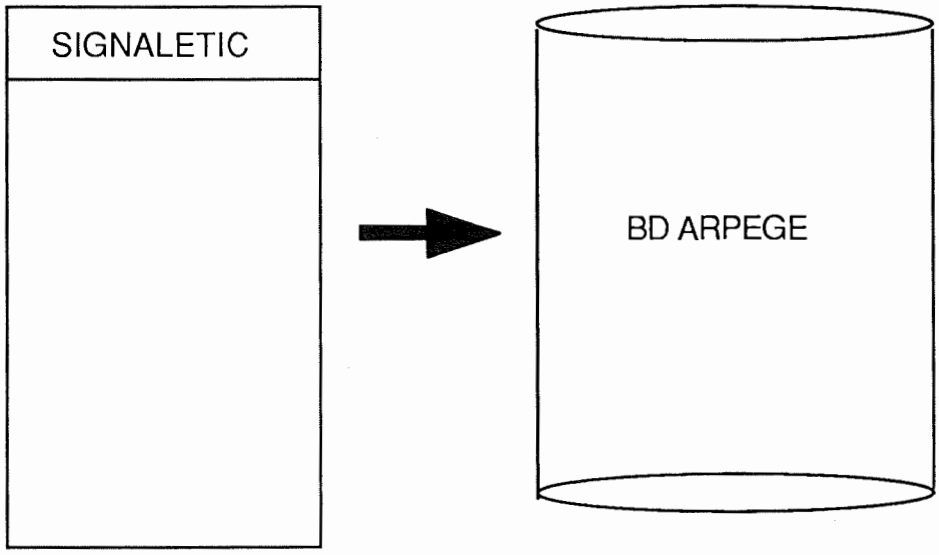
-la conception (dés le début) d'un tel module intermédiaire demande une réflexion concernant le langage utilisé pour sa réalisation car il faut qu'il soit "linkable" avec l'application dont il est question (écrite en Cobol).

-la prise en charge d'un fichier complémentaire doit aussi tenir compte d'un éventuel historique de ce fichier.

## 5. Initialisation de la BD. Arpege.

Cette étape de transition est une étape qui doit être effectuée pour n'importe quelle hypothèse migratoire. Dans ce qui suit nous essayerons, en nous servant des expériences du P.E., de réfléchir en gardant à l'esprit les problèmes qui se poseront à long terme.

### A) Situation de départ:




### B) Préparatifs.

La réalisation de cette phase succède à la phase d'adaptation de la BD Arpege et nécessite la conception(3) de modules de cohérences de niveau 2 et 3 ("le filtre").

Pour chaque zone: réaliser un module de cohérence de niveau 2 (voir le point 6.1. de la Partie 1) et un module de niveau 3 (N3) de manière à tester la plausibilité de la valeur rentrée pour la zone.
Cet ensemble de modules de tests constitue le "**Filtre**" et contribue à un certain niveau de cohérence de la BD.

**P.S.** Parmi les modules que le P.E. avait réalisé, il faut analyser et choisir ceux qu'on pourrait utiliser.

---

(3) Cela a pris un temps important (de l'ordre de plusieurs mois) pour réaliser ces procédures au P.E.

Cette conception est une opération qui demande du temps. Dans le cas du P.E. les tests de niveau 3 étaient les plus restrictifs. Environ 10 % des records passaient sans problèmes de cohérence. De ce fait, le P.E. s'est interrogé sur la recherche d'un équilibre entre la rigueur du filtre et les erreurs qu'on introduit dans la B.D.
Plus on a tendance à faire un filtre moins contraignant, plus on consent à avoir une BD moins cohérente. Il faut donc trouver un compromis.

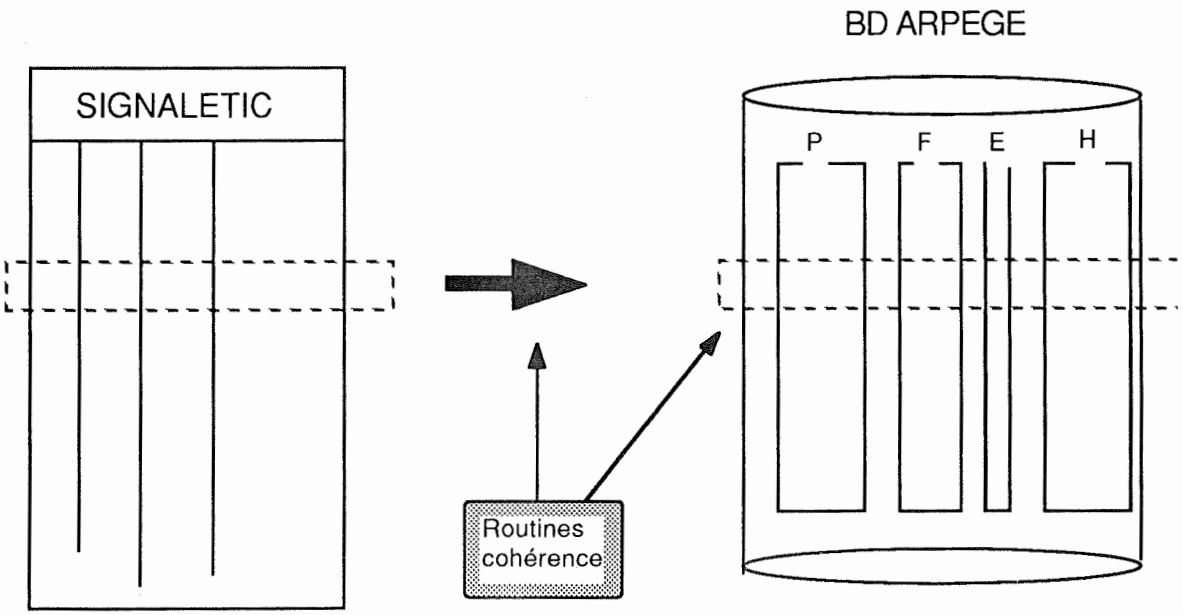
## C) Procédure de chargement.

Pendant la période de chargement de la BD Arpege, on interdit l'accès en écriture dans le Signaletic. Toutes les modifications qu'on voudra faire devront être spécifiées sur feuille. Les feuilles s'accumuleront dans un ordre chronologique.

Les tests de cohérences peuvent s'effectuer:

-soit au moment du chargement, champs après champs pour un record

-soit après chargement dans la BD d'un groupe de records. La deuxième possibilité est plus plus intéressante, car les tests se font entièrement pour un groupe d'enregistrements et émettent dans une liste toutes les incohérences trouvées.



En fonction du nombre de records refusés (si beaucoup), nous rectifierons les modules de cohérence de manière à ce qu'un plus grand nombre de records passent le test tout en ayant un dégré de cohérence acceptable ou satisfaisant.

Il existera des restrictions qui "bloqueront" de nombreux records alors que ceux-ci contribuent peu à la cohérence. Ce sont ces restrictions qu'il faudra ici alléger ou assouplir.
A la fin de cette phase (au P.E. environ 3 semaines), nous prendrons en considération dans la BD Arpege, ainsi initialisée, la pile de modifications qui s'est accumulée jusque là. Ensuite nous apporterons ces modifications dans la BD Arpege.

### D) Phase suivant cette initialisation.

Correction en masse des incohérences introduites sous la phase d'initialisation. Il s'avérera qu'on rencontre des erreurs pour la même zone et pour un grand nombre de records. Il s'agira alors de les corriger. Au P.E., le S.N. a été utilisé pour ce travail de masse.

A ce stade la BD devient plus ou moins fiable, mais son exploitation fera apparaître des erreurs de cohérence (souvent dans l'historique) héritées de la phase d'initialisation, ou plus précisément, de l'assouplissement des routines de cohérences.

# PARTIE 4: DEVELOPPEMENT ET INTEGRATION AU SEIN D'ARPEGE D'UNE FONCTIONNALITE EXISTANTE AU CONSEIL

## 1. Introduction

Au cours de la partie précédente, nous avons réfléchis sur les hypothèses migratoires afin de passer d'un système à un autre. Le problème de l'équivalence des traitements (ou fonctionnalités), a soulevé le cas du manque dans l'Application Arpege, de fonctionnalités nécessaires au Conseil. Dans une telle situation il faut evidemment "adjoindre" à l'application Arpege les fonctionnalités manquantes.
Etant donné que les fonctions des seconde et troisième couches du Conseil resteront en service, même avec l'arrivée d'Arpege, nous nous préoccuperons que des fonctions de la première couche. En effet, ce sont ces dernières qui devront être présentes au sein de l'application Arpege.

Cette quatrième partie va se pencher sur le problème et essayer d'intégrer une des fonctionnalités "manquantes"(1) : GLMENS (une des fonctions de la première couche, voir Partie 1).

On a décrit aussi dans la partie précédente tous les composants (les différents outils, l'historique, ...) de l'application Arpege. Le problème d'intégration d'une nouvelle fonctionnalité va être l'occasion de reparler de tous ces composants. Ceux-ci ne seront plus exposés mais seront utilisés afin de mener à bien la tâche. Aussi, rendra-t-on compte parfois qu'il faut les approfondir (car notre connaissance n'est pas suffisante pour résoudre le problème posé). Ce sera le cas du concept d'historique.
En d'autres termes, on se posera la question du " comment se servir de tous les outils et concepts sous-jacents à l'application Arpege, afin d'intégrer en son sein une nouvelle fonctionnalité ?". Il s'agit en quelque sorte de trouver une méthodologie à suivre de manière à pouvoir y greffer de nouvelles fonctionnalités.

(1) En réalité cette fonctionnalité existe partiellement, et de manière dispersée (dans les répertoires "SuperNatural" propres à chaque utilisateur) au PE.

## 2. La fonction à intégrer.

GLMENS offre les fonctionnalités suivantes (voir Annexes 4 le code source, et en Annexes 5 une partie du listing de son résultat)

:

**(1)** liste des fonctionnaires et agents ayant des enfants à charge qui dépasseront la limite d'âge au cours du mois prochain (limite âge de 18 ou 26 ans si universitaires).

**(2)** liste des fonctionnaires bénéficiant d'un échelon biennal (tous les deux ans d'ancienneté, les fonctionnaires ont droit à l'accroissement automatique de leur échelon d'une unité). L'utilisateur à la possibilité de:

-demander une simple liste

-d'effectuer une mise à jour automatique + listing des enregistrements modifiés

-ou seulement d'effectuer les modifications

**(3)** liste des fonctionnaires bénéficiaires d'un échelon biennal de rémunération: cette fonctionnalité est similaire à la précédente, sauf qu'ici on ne traite que les fonctionnaires qui bénéficient d'une rémunération différente de celle prévue par leur classement (catégorie, grade, échelon)[1]. Donc lorsqu'une personne est promue d'un grade B en un grade A, on doit lui assurer le traitement le plus grand entre celui qu'il avait au grade B, et celui qu'il aura dans le grade A.

**(4)** liste des fonctionnaires ayant des personnes à charge (parents, beaux-parents, ..., à l'exception des enfants) dont la date d'échéance de prise en charge est dépassée ou est proche.

**(5)** liste de tous les fonctionnaires en "Congé de Convenance Personnelle" (CCP), avec pour chacun d'entre eux, ses trois derniers CCP.

**(6)** liste des fonctionnaires ayant des enfants à charge qui vont changer de "code allocation scolaire". Cela concerne les enfants âgés de 11 ans (car ils passent de la maternelle aux études secondaires) et les enfants qui vont atteindre l'âge de 26 ans et qui sont toujours à charge de leurs parents pour cause scolaire ( inscrits dans une Université).

---

[1] Article 46: "Le traitement d'un fonctionnaire ne peut décroître lorsqu'il est promu". Or il existe des postes (les B) qui ont un meilleur traitement qu'un poste mieux classé ( les A). Ceci est du a l'ancienneté du grade. Sans entrer dans les détails les postes se différencient en catégories D, C, B, A. (D < C < B < A). Les postes A sont destinés à des universitaires.

Cette fonction GLMENS fournit donc 6 listes. On peut voir un exemple en Annexes 5.

Ces fonctionnalités de GLMENS ne seront pas toutes prises en considération dans cette intégration. Les objectifs de cette partie sont seulement de donner un exemple d'intégration et de proposer une méthodologie à suivre. On se limitera donc à la prise en considération des fonctionnalités qui offrent un intérêt pour l'analyse. Ainsi la fonctionnalité (3) étant similaire a la (2), on ne s'occupera que de cette dernière.

De plus, bien que cela ne soit pas actuellement indispensable, nous irons jusqu'au codage de certaines fonctionnalités (soit en Natural soit en SN) afin de s'assurer de leur bonne réalisation. Ce sera le cas des fonctionnalités (1), (2), (6). Par contre, pour la (4) on ne donnera que les grandes lignes à suivre.

## 3. Stratégie d'intégration d'une nouvelle fonctionnalité.

### 3.1.Les restrictions de SN

**Si des mises-à-jour sont à effectuer.**

Le langage d'interrogation SN offre la possibilité d'effectuer des travaux de masse qu'il s'agisse de listes (reporting) ou des mises-à-jour. Cet outil etant principalement mis à la disposition des utilisateurs peu expérimentés en informatique, on voit assez mal mettre à leur disposition une fonction de mises-à-jour automatique. Cela peut être dangereux car, par erreur, ils pourraient effectuer des mises-à-jour en masse sans pouvoir s'en rendre compte.

Pour éviter ce genre de problèmes, les concepteurs du projet Arpege ont, dès le début, restreint volontairement les capacités de SN en ne lui donnant que la fonction de reporting. Il est à signaler que cette fonction de reporting est assortie d'opérations (calculs, moyennes, ruptures, ...) assez complexes.

Par conséquent, lorsqu'une fonctionnalité devra effectuer des opérations de mises à jour d'une zone de la BD Arpege, elle ne pourra être effectués au moyen de SN. Elle devra être effectuée via le langage de commande Natural.

**Lorsque la fonctionnalité exigera la recherche de valeurs de champs historisés.**

SN n'as pas la possibilité d'effectuer une recherche dans le fichier historique. Par conséquent, lorsque le programme necessite le traitement du fichier historique, il sera impossible de concevoir la fonctionnalité avec SN. Donc sa réalisation devra être effectuée en Natural.

## 3.2. La démarche à suivre pour la conception de cette fonctionnalité

### A) Identification de champs:

-adaptation de la BD Arpege.

L'une des étapes essentielles avant l'installation d'Arpege au Conseil est l'adaptation de la BD Arpege pour qu'elle contienne les informations (champs) nécessaires au Conseil (voir Partie 3). Puisque cette étape sera réalisée avant l'intégration de nouvelles fonctionnalités, il deviendra superflu de traiter le problème au moment de la conception de fonctions. Dans l'état actuel des choses, étant donné que la normalisation n'a pas encore été effectuée, nous serons obligés de nous y conformer de manière telle que les fonctions désirées puissent être réalisées.

-La correspondance de zones.

Dès que tous les champs du Conseil existeront dans la Bd Arpege, il deviendra utile, avant la réalisation de la fonctionnalité, de voir la correspondance entre l'appellation de la zone dans la "nomenclature Arpege (Annexes 2) et l'appellation de la zone dans le Signaletic. Ceci pour tous les champs utilisés par cette fonctionnalité.

| | Zone 1 | Zone 2 | Zone 3 | | |
|---|---|---|---|---|---|
| Déscription | | | | | |
| Nom au Conseil | | | | | |
| Nom dans la BD Arpege | | | | | |

## B) Interactions avec l'historique.

Comme nous l'avons déjà signalé, nous ne pouvons pas interagir via SN dans l'historique. Afin de remédier à ce besoin assez important, le P.E. a développé une fonction "Exploitation de l'historique".

Au delà des traitements ponctuels des données historisées, on peut avoir des besoins de masse (par exemple pour des statistiques, etc..). C'est dans ce contexte que s'inscrit la fonction "Exploitation de l'historique". Cette fonction permet de rechercher et d'extraire des données de l'historique en fonction de critères choisis par l'opérateur. Le résultat des recherches est sauvegardé dans un fichier spécialement utilisé à cet effet (appélé "subfile") et qui pourra être ultérieurement traité via SN ou même Natural.
Cette fonction se décompose en deux parties, a savoir:

-attribution et gestion des subfiles (pour le stockage des résultats des recherches)

-le traitement des demandes d'extractions de données de l'historique.

### Attribution et gestion des subfiles.

Le gestionnaire de SN pour Arpege dispose d'une transaction qui permet d'associer (ou de supprimer l'usage d') un (des) subfile(s) à un utilisateur donné. Les associations sont sauvées dans le fichier des tables. Par ce biais, on peut à la fois opérer un contrôle d'accès à la fonction exploitation de l'historique, et avoir l'assurance qu'un utilisateur ne pourra pas avoir accès au résultat de la recherche d'un autre (répondant ainsi au souci de sécurité du système). Ce système de gestion comporte donc une série de programmes permettant

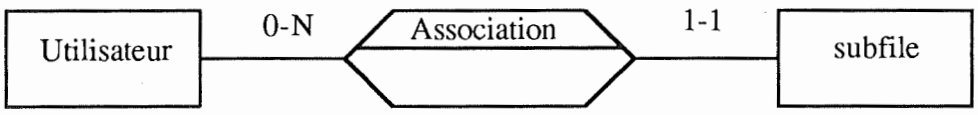
-la saisie (suppression) d'un utilisateur

-la saisie (suppression) d'un subfile

-l'affichage des subfiles déjà attribués

-l'affichage des subfiles associés à un utilisateur donné

-de créer (supprimer) le lien entre un utilisateur et un subfile




### Traitement d'une demande

Les demandes en traitement concernent soit des recherches historiques sur les fichiers de base Personnel ou Emplois (Famille est généralement couplé avec Personnel);

soit des recherches sur les combinaisons Personnel-Famille, Personnel-Emplois, ou Emplois-Personnel.
La reconstitution concerne toujours un enregistrement entier, représentant l'image exacte (situation) de l'objet demandé à la date de reconstitution. Dans le cas d'une combinaison de fichiers, on reconstruit d'abord la situation du fichier maître (ou principal) et ensuite l'image du fichier associé à la même date. Les traitements de reconstitution se font soit à une date donnée, soit entre deux dates fixes (données par l'opérateur), soit entre deux dates "dynamiques" (sur base des valeurs contenues dans deux champs dates du fichier traité, par exemple entre la date d'entrée au Conseil et la date de dernière modification de classement).

Le traitement d'une demande s'opère donc en plusieurs étapes:

1. Phase de sélection (partie on-line)
   - -choix du subfile (ou stocker le résultat)
   - -introduction des critères de sélection de la population
   - -introduction de la date ou période de reconstitution
   - -introduction des champs à reconstituer
   - -introduction des critères de tri des résultats.
2. Phase d'exécution (partie batch)
   - -préparation des critères de sélection
   - -exécution de la sélection
   - -reconstitution des enregistrements sélectionnés à la date ou durant la période demandée.
   - -exécution du tri.

A ce stade l'Exploitation de l'historique nous fournira les résultats dans un subfile connu. Ce dernier pourra être exploité au moyen de SN ou de Natural de manière à fournir les résultats désirés. On verra plus loin comment on se sert de cet outil pour résoudre le cas de la fonctionnalité (5).

### C) La fonctionnalité aura t-elle des mises-à-jour à effectuer?

Dans l'affirmative, on ne pourra pas se servir de SN pour réaliser la fonctionnalité. Il faudra la concevoir au moyen de Natural ou en se servant de la fonction SELECT.

**La fonction SELECT.**

Comme pour le cas de l'historique, le P.E. a développé la fonction SELECT, de manière à remédier au problème des mises-à-jour via SN.

Lorsqu'on choisit de travailler sur le fichier Personnel ou Emplois ou Famille, on a le choix entre:

-Ajout d'un enregistrement
-Effacement d'un enregistrement
-Modification d'un enregistrement
-Affichage d'un enregistrement

Si l'utilisateur choisit une de ces fonctions, le système lui demande un critère de sélection d'enregistrements dans le fichier concerné, c'est l'écran Identification d'enregistrements:

1. Via Num. Matricule
2. Via le Nom
3. Via la fonction SELECT.

Le premier critère de choix se fait moyennant un numéro de matricule (dans ce cas le système affichera un seul enregistrement puisque le numéro de matricule est identifiant). Le second critère de choix concerne les enregistrements ayant la zone Nom égale à une certaine valeur.

Le troisième critère de choix, est la fonction SELECT. Dans ce cas, l'utilisateur formule son critère de recherche en combinant les valeurs des différents champs de l'enregistrement.

```
Ex Select if Zone-1 = xxxx
              and Zone-2 = yyyyy
                    or Zone-3 ....
```

L'utilisateur qui veut effectuer des mises-à-jour (d'un ou plusieurs champs en fonction de la valeur d'un ou plusieurs champs de l'enregistrement) s'y prendra de la manière suivante:

-choix du fichier sur lequel seront effectuées les mises-à-jour. Soit le fichier Personnel ou Emplois. Il n'existe pas de fonction SELECT concernant le fichier Famille.

-choix dans le menu concernant le fichier de l'option "Modification Enregistrement"

-formulation du critère de sélection via la fonction SELECT.

Le système va fournir les enregistrements choisis à l'écran, de manière séquentielle et un à la fois. L'utilisateur n'aura plus qu'à choisir le champs (au moyen de différents "maps") où il aimerait effectuer sa modification.
Cette procédure de mise-à -jour n'est donc pas automatique. Les modifications doivent être effectuées manuellement et une à une. La fonction SELECT ne s'occupe que de l'identification des enregistrements où les modifications doivent êtres faites. Bien que cette procédure soit plus fiable (l'utilisateur agit enregistrement après enregistrement et de manière manuelle), lorsque le nombre de records sélectionnés est assez grand, il devient fastidieux d'effectuer les modifications manuellement. Dans un tel cas, (et si les modifications sont à réaliser **périodiquement**) il devient intéressant de concevoir un programme Natural de manière telle que les modifications soient effectuées en batch. Ainsi l'utilisateur:

-dans un premier temps, demande la liste des enregistrements concernés au moyen de SN

-dans un second temps, exécute le programme qui effectue les mises-à-jour automatiquement et de manière "batch".

Cette dernière solution sera à envisager :

-lorsque les modifications ne concernent pas le fichier Famille, car il n'existe pas de fonction Select pour ce fichier (ce qui est un handicap)[1]

-lorsque le traitement à effectuer est périodique

---

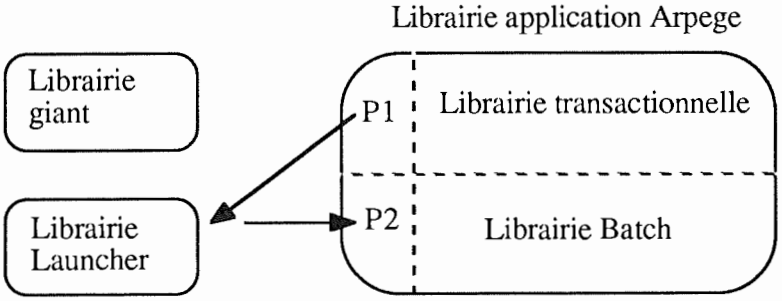
[1] Le PE ne l'avait pas prévue car les besoins en modifications manuelles de ce fichier Famille étaient sporadiques.

-lorsque le nombre de records susceptibles d'être sélectionnés est important (en fonction de la patience de la personne).

Néanmoins, ce programme en Natural devra obéir aux deux règles suivantes:

-qui dit mises-à-jour, dit historisation des anciennes valeurs. Dans ce cas, le programme doit se charger d'historiser les anciennes valeurs. Alors que dans le cas de l'utilisation de la fonction SELECT, le système se chargeait de cette tâche

-ce programme va se décomposer en deux parties:




-un programme P1 en mode transactionnel. Ce dernier se chargeant de:

* prendre note de ce que l'utilisateur demande
* lancer le programme batch P2 via le Launcher.

-un autre programme, P2, qui travaillera en batch et qui sera lancé par la Launcher.

## 3.3. Intégration de cette fonction au sein de la librairie Arpege

Si la fonction à été réalisée au moyen de la fonction SELECT, le problème d'intégration ne se pose pas. Il en est de même lorsque la fonction a été réalisée via l'outil SN. En effet, dans ce cas la fonction (qui sera appelé "transaction SN") sera stockée dans la directory des programmes SN de la personne. L'utilisateur l'exécutera lorsqu'il en aura besoin.
Dans le cas où la fonction à été réalisée au moyen de Natural, il faudra intégrer le programme parmi les programmes ad-hoc correspondant au fichier concerné. Par exemple si le programme effectue des modifications dans le fichier Personnel, il faudra

l'insérer dans la liste des programmes ad-hoc du menu du Personnel. De plus, afin de gérer la sécurité du programme, il faudra entrer dans la fonction Giant et donner les attributs d'utilisation (lire, modifier, supprimer, ...) pour chaque personne (ou groupe de personnes), c'est-à-dire associer le programme avec ses utilisateurs.

## 4. Développement des fonctionnalités.

Comme on vient de le voir, l'Application Arpege propose des outils et des restrictions lorsque l'utilisateur veut concevoir un programme. La fonctionnalité GLMENS sera pour cette raison "splittée", divisée, de manière telle qu'on puisse utiliser les outils mis à notre disposition. Pour chaque sous-fonctionnalité, on fera soit une transaction SN soit un programme Natural.

### 4.1. Enfants dépassant la limite d'âge.

*A) Enoncé*: liste des fonctionnaires ayant des enfants qui vont atteindre l'âge de 18 ou 26 ans.

*B) Identification des champs*

| | | | | | |
|---|---|---|---|---|---|
| Appellation | Nom fonctionnaire | Prénom fonctionnaire | Matricule fonctionnaire | Prénoms enfants | Date de nais. enfants |
| Dans Isdam | Signaletic. Nom | Signaletic. Prenom | Signaletic. Nrmatr | Signaletic. Epre | Signaletic. Edna |
| Dans Arpege | Fg-Pers.Nom | Fg-Pers. prenom | Fg-Pers. Nopers | Fg-Fami. pacprn | Fg-Fami. pacdna |

Pour les zones du fichier Isdam voir Annexes 1.

*B) Intéractions avec l'historique.*
La sous-fonctionnalité ne demande pas des données de l'historique.

*C) Mises-à-jour à effectuer.*

La sous-fonctionnalité ne fait que consulter les données, elle n'effectue pas de mises-à-jour.

Par conséquent, la réalisation peut être effectuée au moyen de SN. Pour plus de complétude on divisera la sous-fonctionnalité en deux transactions SN. L'une, T1a, s'occupant des enfants de 18 ans, l'autre, T1b, pour des enfants de 26 ans.

*D) Les transactions SN: voir ci-après T1a et T1b*

Dans le *select* on sélectionne les enfants qui auront l'âge recherché et on choisit les champs nécessaires. Ensuite dans les *Logic statements* on affine le critère de sélection: il faut que les fonctionnaires intéressés

- soient des fonctionnaires titulaires (LSTTYPC = 1)
- soient en activité, ou suspension momentanée ou partielle (POSTYPC < 30)
- aient l'enfant réellement à charge (CHGTYP <>décédé ...)

La dernière page du *reporting* donne le résultat trouvé.

T1a Enfants de 18 ans

```
Select: 1  PACREF > '090000' AND PACTYP = 'ENF' AND PACDNA > 19730228
        2  AND PACDNA < 19730401                                          more N
Ref Dis S/C Func   Field Name          Ref Dis S/C Func   Field Name
 AE                K .PACREF           LX
 AI 4        >        .PACPRN          LY
 AJ                   .PACTYP          LZ
 AK 5                 .PACDNA          MA
 DG 1        >     K .NOPERS           MB
 FB 2        >     K .NOM              MC
 FC 3        >        .PRENOM          MD
 FR                K .LSTTYPC          ME
 GC                K .POSTYPC          MF
 LT                                    MG
 LU                                    MH
 LV                                    MI
 LW                                    MJ

Select. lines; ST=top  SU=up  SD=down  SB=bottom  SI=insert       more fields: N
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
cont  help  cmd   end                 top   up    down  bot   save  run   main
```

**Logic statement** 1 of 2

IF Condition:

LSTTYPC NE 1 OR POSTYPC > 30

THEN Action:

REJECT

ELSE Action:

```
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9----PF10--PF11--PF12---
cont  help  cmd   end         add   top   up    down  bot   save  run   main
```

**Logic** statement 2 of 2

IF Condition:

CHGTYP = 'DCD' OR CHGTYP = 'FIN' OR CHGTYP = '2CF'

THEN Action:

REJECT

ELSE Action:

```
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
cont  help  cmd   end         add   top   up    down  bot   save  run   main
```

CM→Entr 1

07/05/91

| NO.PERS. | NOM TIT. | PRENOM | PRENOM ENF. | DATE NAISS. |
|---|---|---|---|---|
| 095187 | HICKEY | JOSEPH | PAMELA | 19730306 |
| 095247 | DIETZ | MICHEL | DANIELLE | 19730328 |
| 095346 | PIROTTE-GERARD | MARGUERITE | PATRICK | 19730305 |

*** End of report ***

Number of records processed: = 3

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
 own print mod top down main

T1b ENFANTS 16 ans.

```
Select: 1  PACREF > ' ' AND PACTYP = 'ENF' AND PACDNA > 19640228
        2  AND PACDNA < 19640401                                          more N
Ref Dis S/C Func   Field Name          Ref Dis S/C Func   Field Name
 AE                K .PACREF           LX
 AI 4        >        .PACPRN          LY
 AJ                   .PACTYP          LZ
 AK 5                 .PACDNA          MA
 DG 1        >     K .NOPERS           MB
 FB 2        >     K .NOM              MC
 FC 3        >        .PRENOM          MD
 FR                K .LSTTYPC          ME
 GC                K .POSTYPC          MF
 LT                                    MG
 LU                                    MH
 LV                                    MI
 LW                                    MJ

Select. lines: ST=top  SU=up  SD=down  SB=bottom  SI=insert       more fields: N
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
cont  help  cmd   end                 top   up    down  bot   save  run   main
```

Logic statement 1 of 2

IF Condition:

LSTTYPC NE 1 OR POSTYPC > 30

THEN Action:

REJECT

ELSE Action:

```
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
cont  help  cmd   end         add   top   up    down  bot   save  run   main
```

Logic statement 2 of 2

IF Condition:

CHGTYP = 'DCD' OR CHGTYP = 'FIN' OR CHGTYP = '2CF'

THEN Action:

REJECT

ELSE Action:

```
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
cont  help  cmd   end         add   top   up    down  bot   save  run   main
```

CM-ENF2

1

07/05/91

| NO.PERS. | NOM TIT. | PRENOM | PRENOM ENF. | DATE NAISS. |
|---|---|---|---|---|
| 069268 | VANDEWOESTIJNE | AGNES | JOHAN | 19640238 |

*** End of report ***

Number of records processed: = 1

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
Down print mod top down main

## 4.2. Echéance des personnes à charge.

*A) Enoncé:* donner la liste des fonctionnaires ayant des personnes à charge (parents, ...) dont la date d'échéance est toute proche (dans le mois).

*B) Identification des champs*

| Appellation | Matricule fonct. | Nom fonct. | Prenom fonct. | Nom-Prén. pers charge | Lien de parenté | Date de naissance | |
|---|---|---|---|---|---|---|---|
| Dans Isdam | Signaletic. Nrmatr | Signaletic. Nom | Signaletic. Prenom | Signaletic. Pcnom & Pcpre | Signaletic. Pcpar | Signaletic. Pcdna | |
| Dans Arpege | Fg-Pers. Nopers | Fg-Pers. Nom | Fg-Pers. Prenom | Fg-Fami. Pacnom & Pacprn | Fg-Fami. Pactyp | Fg-Fami. Pacdna | |

| | Début decision | Echéance decision | Alloc. foyer | Code résidence |
|---|---|---|---|---|
| | Signaletic. Pcdeb | Signaletic. Pcech | Signaletic. Pcetcv | Signaletic. Cres |
| | Fg-Fami. Chgeff | Fg-Fami. Chgech | Fg-Fami Chgtyp | Fg-Fami. Restyp |

*C) Intéractions avec l'historique*

Aucune demande de consultations des données de l'historique.

*D) Demandes de mises-à-jour*

Pas de modifications des données de la BD Arpege à effectuer. C'est du simple reporting.

Par conséquent la réalisation sera effectuée au moyen de SN.

*E) La transaction: voir ci-après T2*

Dans un premier temps, on sélectionne via le *Select,* les personnes à charges qui répondent à notre critère, c'est-à-dire:

- le lien de parenté PACTYP = 'ASS' (il faut que soit des assimilés et non des enfants)
- la date d'echéance CHGECH étant proche.

# T2 : Echéance personnes à charge

```
Select: 1  PACREF > '094000' AND CHGECH > 19910315 AND
        2  CHGECH < 19910415 AND PACTYP = 'ASS'                    more N
Ref Dis S/C Func   Field Name              Ref Dis S/C Func   Field Name
 AA                G HISTORY-SYSTEM-FIE    CE 8        >       .CHGEFF
 AB                  .IDENT                CF 9        >       .CHGECH
 AC                G NON-HISTORISED-FIE    CG 11       >       .RESTYP
 AD                  .HSPEFF               DC                G HISTORY-SYSTEM-FIE
 AE 1        >     K .PACREF               DD                  .IDENT
 AF                  .PACNUM               DE                G NON-HISTORISED-FIE
 AG                  .PACSEX               DF                  .HSTEFF
 AH 4        >       .PACNOM               DG                K .NOPERS
 AI 5        >       .PACPRN               EZ                G HISTORISED-FIELDS
 AJ                  .PACTYP               FB 2              K .NOM
 AK 7        >       .PACDNA               FC 3                .PRENOM
 CC                G HISTORISED-FIELDS     FR                K .LSTTYPC
 CD 10       >     K .CHGTYP               GC                K .POSTYPC

Select. lines: ST=top  SU=up  SD=down  SB=bottom  SI=insert      more fields: N
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
cont  help  cmd   end               top   up    down  bot   save  run   main
```

Logic statement 1 of 2

IF Condition:

CHGECH > 19910315 AND CHGECH < 19910415

THEN Action:

ACCEPT

ELSE Action:

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
cont help cmd end add top up down bot save run main

Logic statement 2 of 2

IF Condition:

LSTTYPC GT 7 OR POSTYPC GT 30

THEN Action:

REJECT

ELSE Action:

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
cont help cmd end add top up down bot save run main

CM-3



07/05/91

| MAT.TIT. | NOM TIT. | PRENOM | NOM PAC | PRENOM | DTE-NAIS | CHARGE | DEBUT | ECHEANCE | RES |
|---|---|---|---|---|---|---|---|---|---|
| 096149 | TABAKA | MONIKA | HOFACKER | PAULA | 19170309 | ALL | 19900401 | 19910331 | |

*** End of report ***

90 C.

De plus on choisit les champs qui nous intéressent. Dans les *Logic statements* on affine le critère de sélection (concernant le lien statutaire et la position administrative du fonctionnaire concerné).

### 4.3 Echelon biennal

*A) Enoncé:* Liste des fonctionnaires bénéficiaires d'une augmentation de leur échelon (toutes les deux années)

*B) Identification des champs*

| Appellation | Matricule fonct. | Nom fonct. | Prenom fonct. | Grade fonct. | Ancienneté grade | Echelon | Ancienneté echelon |
|---|---|---|---|---|---|---|---|
| Dans Isdam | Signaletic. Nrmatr | Signaletic. Nom | Signaletic. Prenom | Signaletic. Grade | Signaletic. Dgrade | Signaletic. Echel | Signaletic. Dechel |
| Dans Arpege | Fg-Pers. Nopers | Fg-Pers. Nom | Fg-Pers. Prenom | Fg-Pers. Clsgra | Fg-Pers. Clsagr | Fg-Pers. Clsstp | Fg-Pers. Clsast |

*C) Interactions avec l'historique.*

Aucune nécessité de consultation de l'historique.

*D) Mises-à-jour à effectuer.*

La sous-fonctionnalité aura à effectuer des modifications dans la BD Arpege. Ces modifications concernent l'avancement de l'échelon du fonctionnaire bénéficiaire:

(échelon CLSSTP <-- CLSSTP + 1)

Conséquences de C) et D) : la sous-fonctionnalité devra être effectuée soit

- au moyen de la fonction SELECT
- avec Natural

Si on décide de la réaliser au moyen du SELECT, les inconvénients sont les suivants:

- vu que la sous-fonctionnalité sera très souvent utilisée (tous les mois), on devra chaque fois l'effectuer à la main. Ce qui deviendrait, à la longue, assez fastidieux

- vu que le nombre de fonctionnaires sélectionnés est assez grand, le faire à la main serait une procédure longue et fatigante.

Si on décide de la concevoir au moyen de Natural, on rencontre l'inconvénient de devoir traiter le problème d'historisation des zones modifiés (alors que dans le SELECT il est automatiquement pris en charge)

*E) Le programme Natural*

Le programme Natural va se décomposer en deux parties:

- la première partie *CM-ECHTP* : qui se chargera de faire l'interface "on-line" avec l'utilisateur. Ce dernier lancera, via le Launcher, le programme batch *CM-ECHBT*. Il se situera dans les modules de la partie on-line de l'application Arpege.
- la seconde partie CM-ECHBT : programme batch s'occupant de répondre aux besoins de la sous-fonctionnalité. C'est la partie qui réalise la sélection et la modification dans la BD Arpege.

F) *CM-ECHTP*: il se compose des parties suivantes:

| |
|---|
| Définition des données:<br>-RJE-Parameter: ce sont les paramètres qui feront le lien entre le programme "on-line" et le programme "batch" |
| Demander à l'utilisateur son choix:<br>liste, mise-à-jour, ou mise-à-jour avec liste |
| Lancement du programme batch:<br>CALLNAT 'LAUNCHER' RJE-PARAMETER |
| Contrôler si le lancement s'est bien déroulé:<br>on verifie le résultat de la zone RJE-RETURN |

```
)EXT LIST CM-ECHTP
0010 ************************************************************************
0020 *                                                                      *
0030 * NOM : CM-ECHTP                                                       *
0040 * ===                                                                  *
0050 *                                                                      *
0060 * OBJET                                                                *
0070 * =====                                                                *
0080 *        LANCEMENT BATCH GESTION DES LISTES AVANCEMENT ECHELON.         *
0090 *                                                                      *
0100 ************************************************************************
0110 *                                                                      *
0120 * FONCTION :  PROGRAMME DE LANCEMENT EN BATCH DES FOCTIONNAIRES        *
0130 * ==========  BENEFICIAIRES D'UN ECHELON BIENNAL                        *
0140 *                                                                      *
0150 * LIENS HIERARCHIQUES :                                                *
0160 * =====================                                                *
0170 *       PGM.APPELANT : MDRIVER (VIA COMMANDES)                         *
0180 *      SOUS-FONCTIONS: LAUNCHER                                        *
0190 *                                                                      *
0200 *       MAPS APPELES : /                                               *
0210 *                                                                      *
0220 ************************************************************************
0230 *
0240 DEFINE DATA LOCAL
0250 * ***************
0260 *
0270 1 RJE-PARAMETERS
0280   2 RJE-REQUEST-NUMBER          (A4)
0290   2 RJE-TSN-NUMBER              (N4)
0300   2 RJE-TIME-LIMIT              (N4)
0310   2 RJE-DATABASE-ID             (N1)
0320   2 RJE-APPLICATION-ID          (A8)
0330   2 RJE-PROGRAM                 (A8)
0340   2 RJE-USER-ID                 (A8)
0350   2 RJE-USER-PASSWORD           (A8)
0360   2 RJE-FORM-TYPE               (A4)
0370   2 RJE-DESTINATION             (A4)
0380   2 RJE-IMMEDIATE-ENTER         (A1)
0390   2 RJE-SYSLST                  (A1)
0400   2 RJE-NUMBER-OF-COPIES        (N1)
0410   2 RJE-NUMBER-OF-WORKFILES     (N1)
0420   2 RJE-NUMBER-OF-PRINTFILES    (N1)
0430   2 RJE-DATA                    (A80)
0440   2 RJE-RETURN-CODE             (N2)
0450 *
0460 1 #ERR-TEXT(A60)
0470 1 #ETAT     (A30)
0480 1 #NCOPIES (N1)
0490 *
0500 1 #DATA     (A80)
0510 1 REDEFINE #DATA
0520   2 #CHOIX (A1)
0530   2 #FILLER(A79)
0540 *
0550 END-DEFINE
0560 *
0570 * ---------------------------------------------------------------------
0580 *
0590 RELEASE STACK
```

```
0600 RESET #DATA
0610 *
0620 INPUT   10X 'MENU SELECTION' /
0630         10X '-' (17) //
0640         10X '1. SIMPLE LISTE' /
0650         10X '2. MISE A JOUR SANS LISTE' /
0660         10X '3. MISE A JOUR AVEC LISTE' /
0670         10X ' . QUITTER' ///
0680         10X 'ENTREZ LA SELECTION :' #CHOIX
0690 IF #CHOIX = '1' OR = '2' OR = '3' OR = ' '
0700    IGNORE      /* O.K.
0710 ELSE
0720    REINPUT 'CHOIX ERRONE !!!'
0730 END-IF
0740 *
0750 *
0760 *  LANCEMENT BATCH...
0770 *
0780 MOVE 'CM-ECHBT' TO RJE-PROGRAM
0790 MOVE 1          TO RJE-NUMBER-OF-PRINTFILES
0800 MOVE 60         TO RJE-TIME-LIMIT
0810 MOVE 'Y'        TO RJE-IMMEDIATE-ENTER
0820 MOVE 'Y'        TO RJE-SYSLST
0830 MOVE #DATA      TO RJE-DATA
0840 MOVE 'SITE'     TO RJE-DESTINATION                    /* 'LBAK' = LASER
0850 * MOVE 'LASA'   TO RJE-FORM-TYPE                      /* FORMAT LASER
0860 *
0870 CALLNAT 'LAUNCHER' RJE-PARAMETERS
0880 *
0890 IF RJE-RETURN-CODE NE 0
0900    INPUT USING MAP 'H#EBATCH'
0910    FETCH 'MDRIVER' #ERR-TEXT
0920 ELSE
0930    COMPRESS 'LST.PROD.LIST' RJE-REQUEST-NUMBER INTO #ETAT LEAVING NO
0940    SET KEY PF20
0950    SET CONTROL 'WFL53C1085/14<<--'
0960    INPUT USING MAP 'H#GLOSER'
0970    SET CONTROL 'WML79C238<<--'
0980    FETCH 'MDRIVER' '0131'
0990 END-IF
1000 *
1010 *
1020 *
1030 END
***** End of List *****
```

G) *CM-ECHBT*:

Le critère de sélection :

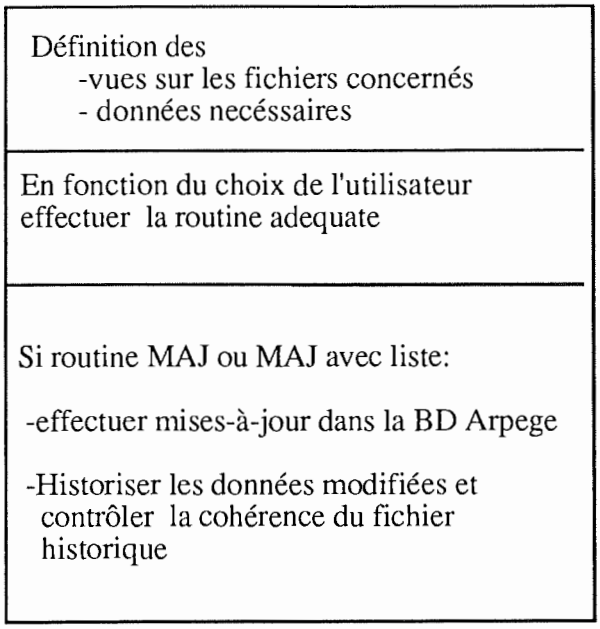
-fonctionnaires ayant la position administrative POSTYPC < 24 (on ne prend pas ceux qui sont en CCP ou qui sont détachées, ...)

-fonctionnaires ayant une ancienneté de l'échelon (CLSAST) égale à deux années.

-fonctionnaires pouvant être "promus". En fonction du grade d'une personne les valeurs que peut prendre l'échelon sont différentes. Ainsi, par exemple, l'échelon des grades A3, A4 (et d'autres) peut aller jusqu'à une valeur de huit (borne supérieure"). L'échelon des grades A1, A2 (et d'autres) peut aller jusqu'à six. L'échelon du grade A8 a un échelon peut aller jusqu'à deux. On comprend donc que les fonctionnaires concernés par une promotion (de manière automatique) sont ceux pour qui, en fonction du grade, on peut accroître l'échelon d'une unité. Ceux qui ont l'échelon à la "borne supérieure" ne peuvent pas être pris. A cet effet, on a conçu un petit programme, CM-ECHEL, qui indique (en fonction du grade et de l'échelon) si un fonctionnaire peut être "échelonné" ou pas. Ce programme est le suivant:

```
NEXT LIST CM-ECHEL
0010 ************************************************************************
0020 *                                                                      *
0030 * NOM : CM-ECHEL     DATE ECRIT. :  10/05/91     AUTEUR : C. SAITTA    *
0040 *                                                                      *
0050 * OBJET                                                               *
0060 * ==========                                                          *
0070 *      VERIFIE SI UNE PERSONNE, EN FONCTION DE SON CLASSEMENT          *
0080 *      EST PROMOUVABLE                                                 *
0090 *                                                                      *
0100 *                                                                      *
0110 * LIENS HIERARCHIQUES :                                               *
0120 * ====================                                                *
0130 *    CALLED BY       : CM-ECHBT                                        *
0140 *                                                                      *
0150 ************************************************************************
0160 *
0170 DEFINE DATA PARAMETER
0180 *
0190 1 #GRADE            (A2)                  /* INPUT
0200 1 #ECHELON           (N1)                  /* INPUT
0210 1 #ECHELONNABLE      (L)                   /* OUTPUT
0220 *
0230 END-DEFINE
0240 *
0250 MOVE FALSE TO #ECHELONNABLE
0260 *
0270 IF (#GRADE = 'A3' OR = 'A4' OR = 'A5' OR = 'A6' OR = 'B1' OR = 'B2'
0280             OR = 'B3' OR = 'B4' OR = 'C1' OR = 'C2' OR = 'C3' OR = 'C4'
0290             OR = 'D1' OR = 'D2' OR = 'D3'
0300             OR = 'L3' OR = 'L4' OR = 'L5' OR = 'L6')
0310    AND (#ECHELON < 8)
0320    MOVE TRUE TO #ECHELONNABLE
0330 END-IF
0340 *
0350 IF (#GRADE = 'A1' OR = 'A2' OR = 'A7' OR = 'L7')
0360    AND (#ECHELON < 6)
0370    MOVE TRUE TO #ECHELONNABLE
0380 END-IF
0390 *
0400 IF (#GRADE = 'B5' OR = 'C5' OR ='D4')
0410    AND (#ECHELON < 5)
0420    MOVE TRUE TO #ECHELONNABLE
0430 END-IF
0440 *
0450 IF (#GRADE = 'A8') AND (#ECHELON < 2)
0460    MOVE TRUE TO #ECHELONNABLE
0470 END-IF
0480 *
0490 END
***** End of List *****
```

Le programme CM-ECHBT se compose des parties suivantes:




Le programme est le suivant:

```
NEXT LIST CM-ECHBT
0010 ************************************************************************
0020 * PROGRAMME: CM-ECHBT                                                  *
0030 * OBJET : EFFECTUE LA PARTIE BATCH DU PROGRAMME AYANT                  *
0040 *         COMME BUT DE GERER LES FONCTIONNAIRES QUI ONT                *
0050 *         DROIT A UN ECHELLON.                                         *
0060 * DATE ECR. 10-05-91                                                   *
0070 * AUTEUR: C. SAITTA                                                    *
0080 *                                                                      *
0090 * IL APPELLE :  N#HSTUPD, CM-ECHEL                                     *
0100 *                                                                      *
0110 * APPELLE PAR : VIA LE LAUNCHER PAR CM-ECHTP                           *
0120 *                                                                      *
0130 ************************************************************************
0140 *
0150 DEFINE DATA LOCAL
0160 *
0170 1 PERSONNEL-VIEW VIEW OF FG-PERS
0180   2 NOPERS
0190   2 NOM
0200   2 PRENOM
0210   2 CLSGRA
0220   2 CLSAGR
0230   2 CLSSTP
0240   2 CLSAST
0250   2 POSTYPC
0260 ****** POUR L'HISTORISATION ... ************
0270   2 IDENT
0280   2 MAJEFF
0290   2 MAJMODC
0300   2 MAJDEC
0310   2 CLSEFF
0320 *
0330 1 #PARAM (A80)
0340 *
0350 1 REDEFINE #PARAM
0360   2 #CHOIX (A1)
0370   2 #FILLER(A79)
0380 *
0390 1 #DATE-CALC(N8)
0400 1 #DATE-DEB (N8)
0410 1 #DATE-FIN (N8)
0420 1 #DATE-NECH(N8)
0430 1 ##NUM-ERR (N4)
0440 1 #PROMU        (L)
0450 *
0460 *******    PARAMETRES POUR L'HISTORISATION ...
0470 *
0480   1 #FILENAME   (A1) INIT <'P'>
0490   1 #IDENT     (B4)
0500   1 #DATE      (N8)
0510   1 #FIELDNAME (A32)
0520   1 #FIELDVALUE (A50)
0530   1 #MODCOD    (N3)
0540   1 #DATDEC    (N8)
0550   1 #USER-OPER (A6)
0560   1 #RESULT    (L)
0570   1 #MAJ-SF    (L)
0580 *
0590   1 HIST VIEW OF FG-HISTORY
```

```
0600      2 FIELDNAME
0610      2 DATE
0620      2 MODCOD
0630      2 DATDEC
0640    1 #SUP-IDF     (A26)
0650    1 REDEFINE #SUP-IDF
0660      2 #SFILE     (A1)
0670      2 #SIDENT    (B4)
0680      2 #SDATE     (N8)
0690      2 #SFIELD    (A10)
0700 ********** POUR LA GESTION COHERENTE DE L'HISTORIQUE ...
0710    1 #W-DATE(A8)
0720    1 #SW-WARNERR(L)
0730 END-DEFINE
0740 *
0750 ****** FIN DE DECLARATION DES DONNEES ***************************
0760 *
0770 FORMAT LS=130
0780 FORMAT (1) PS=55 LS=132
0790 * ************************************************************
0800 INPUT #PARAM
0810 * ************************************************************
0820 MOVE *DATN TO #DATE-DEB                    /* DEBUT SELECTION MOIS
0830 COMPUTE #DATE-FIN  = #DATE-DEB + 30        /* FIN SELECT. MOIS (+30 JOURS)
0840 COMPUTE #DATE-NECH = #DATE-DEB + 20000     /* NOUVELLE ANC. ECH.
0850 *
0860 * ************************************************************
0870 DECIDE ON FIRST VALUE OF #CHOIX
0880        VALUE '1'    PERFORM LISTE
0890        VALUE '2'    PERFORM MAJ
0900        VALUE '3'    PERFORM MAJ-LISTE
0910        NONE VALUE   TERMINATE
0920 END-DECIDE
0930 *
0940 ********** LES TROIS ROUTINES ...*****************************
0950 *** 1. SI ON NE DEMANDE QU'UNE LISTE...
0960 *
0970 DEFINE SUBROUTINE LISTE
0980 *
0990 READ PERSONNEL-VIEW
1000      WHERE    POSTYPC < 24   AND
1010               (CLSAST GE #DATE-DEB AND  CLSAST LE #DATE-FIN)
1020  CALLNAT 'ECHELON'    CLSGRA    CLSSTP #PROMU
1030  IF #PROMU
1040    DISPLAY NOTITLE NOPERS NOM PRENOM CLSGRA CLSAGR CLSSTP CLSAST
1050  END-IF
1060 END-READ
1070 END-SUBROUTINE
1080 *
1090 ************************************************************
1100 ** 2. SI ON DEMANDE UNE SIMPLE MISE-A-JOUR....
1110 *
1120 DEFINE SUBROUTINE MAJ
1130  READ PERSONNEL-VIEW
1140                    WHERE POSTYPC < 24
1150                    AND   (CLSAST GE #DATE-DEB  AND CLSAST LE #DATE-FIN)
1160  CALLNAT 'CM-ECHEL'  CLSGRA  CLSSTP #PROMU
1170   IF #PROMU
1180 ***** ALORS ON EFFECTUE LES MISES-A-JOURS ... **********************
1190      COMPUTE CLSAST = CLSAST + 20000
```

```
1200       COMPUTE CLSSTP = CLSSTP + 1
1210       UPDATE
1220 ***** ET ON HISTORISE LES VALEURS DANS LE FICHIER HISTORIQUE ... *
1230       PERFORM HISTO
1240       IF ##NUM-ERR NE 0
1250          BACKOUT TRANSACTION
1260       END-IF
1270    END-IF
1280    END TRANSACTION
1290  END-READ
1300 END-SUBROUTINE
1310 *
1320 ************************************************************
1330 * 3. SI ON DEMANDE MISE-A-JOUR ET LISTE...
1340 *
1350 DEFINE SUBROUTINE MAJ-LISTE
1360 *
1370 READ PERSONNEL-VIEW WHERE POSTYPC < 24  AND
1380                          (CLSAST GE #DATE-DEB AND CLSAST LE #DATE-FIN)
1390  CALLNAT 'CM-ECHEL' CLSGRA CLSSTP #PROMU
1400  IF #PROMU
1410       DISPLAY NOTITLE NOPERS NOM PRENOM CLSGRA CLSAGR CLSSTP CLSAST
1420 **********  ON EFFECTUE LES MISES-A-JOUR... **************************
1430       COMPUTE CLSAST = CLSAST + 20000
1440       COMPUTE CLSSTP = CLSSTP + 1
1450       UPDATE
1460 **********  ON HISTORISE LES VALEURS MODIFIEES ********************
1470       PERFORM HISTO
1480       IF ##NUM-ERR NE 0
1490          BACKOUT TRANSACTION
1500       END-IF
1510  END-IF
1520  END TRANSACTION
1530 END-READ
1540 END-SUBROUTINE
1550 ************************************************************************
1560 *              POUR L'HISTORISATION                                    *
1570 ************************************************************************
1580 DEFINE SUBROUTINE HISTO
1590 *
1600 *  INITIALISATIONS PARAMETRES HISTORIQUE ...
1610 *
1620 MOVE PERSONNEL-VIEW.IDENT TO       #IDENT
1630 MOVE PERSONNEL-VIEW.MAJEFF TO      #DATE
1640 MOVE PERSONNEL-VIEW.MAJMODC TO     #MODCOD
1650 MOVE PERSONNEL-VIEW.MAJDEC TO      #DATDEC
1660 MOVE *USER TO                     #USER-OPER
1670 RESET #RESULT #MAJ-SF ##NUM-ERR
1680 MOVE FALSE TO #SW-WARNERR
1690 **** FIN INITIALISATIONS VARIABLES ...
1700 *
1710 **** HISTORISATION SUR DATE EFFET PROPRE ...
1720 *
1730 MOVE PERSONNEL-VIEW.CLSEFF TO      #FIELDVALUE
1740 MOVE 'CLSEFF' TO                   #FIELDNAME
1750 PERFORM CALL-HISTORY
1760 IF ##NUM-ERR NE 0
1770    ESCAPE ROUTINE
1780 END-IF
1790 *
```

```
1800 **** HISTORISATION DE L'ECHELON,  CLSSTP...
1810 *
1820 MOVE PERSONNEL-VIEW.CLSSTP TO      #FIELDVALUE
1830 MOVE 'CLSSTP' TO                   #FIELDNAME
1840 PERFORM CALL-HISTORY
1850 IF ##NUM-ERR NE 0
1860    ESCAPE ROUTINE
1870 END-IF
1880 *
1890 **** HISTORISATION DE L'ANCIENNETE DE L'ECHELON,  CLSAST...
1900 *
1910 MOVE PERSONNEL-VIEW.CLSAST TO      #FIELDVALUE
1920 MOVE 'CLSAST' TO                   #FIELDNAME
1930 PERFORM CALL-HISTORY
1940 IF ##NUM-ERR NE 0
1950    ESCAPE ROUTINE
1960 END-IF
1970 *
1980 ****   WARNING COHERENCE HISTORIQUE ...
1990 IF #SW-WARNERR
2000    MOVE 9999 TO ##NUM-ERR
2010 END-IF
2020 END-SUBROUTINE
2030 *
2040 **** SUBROUTINE D'HISTORISATION  CALL-HISTORY
2050 ************************************************
2060 DEFINE SUBROUTINE CALL-HISTORY
2070  CALLNAT 'N#HSTUPF' #FILENAME #IDENT #DATE #FIELDNAME #FIELDVALUE
2080                     #MODCOD #DATDEC #USER-OPER #RESULT #MAJ-SF
2090 *
2100 **** #RESULT ET #MAJ-SF SONT LES RESULTATS FOURNIS PAR CETTE PROCEDURE
2110 ***  ON VA LES TESTER ...
2120 *
2130 IF NOT #RESULT
2140    BACKOUT TRANSACTION             /* PROBLEME SYSTEME...
2150    WRITE 'ERREUR GRAVE DU SYSTEME...CONTACTEZ LE DBA.'
2160    MOVE 123 TO ##NUM-ERR
2170    ESCAPE ROUTINE
2180 END-IF
2190 *
2200 IF NOT #MAJ-SF
2210  IF #FIELDNAME NE 'MAJEFF'
2220    BACKOUT TRANSACTION             /* PROBLEME COHERENCE HISTORIQUE...
2230    WRITE 'ERREUR HISTORIQUE...NON SITUATION FINALE...'
2240    MOVE 123 TO ##NUM-ERR
2250    ESCAPE ROUTINE
2260    END-IF
2270 END-IF
2280 END-SUBROUTINE                     /* CALL-HISTORY
2290 *  ************************************************************
2300 END
***** End of List *****
```

### 4.4. Echéances CCP (Congés de Convenance Personnelle)

*A) Enoncé* : donner la liste des gens qui se trouvent en CCP et pour chacun d'eux les trois derniers CCP pris avec date de début et la date de fin.

*B) Identification des champs:*

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Appellation | Matricule fonct. | Nom fonct. | Prenom fonct. | Grade | Nationnal. | CCP actuel Du - Au | |
| Dans Isdam | Signaletic. Nrmatr | Signaletic. Nom | Signaletic. Prenom | Signaletic. Grade | Signaletic. Natacl | Signaletic. Cdsd & Cdsf | |
| Dans Arpege | Fg-Pers. Nopers | Fg-Pers. Nom | Fg-Pers. Prenom | Fg-Pers Clsgra | Fg-Pers. Etcnat | Fg-Pers. Posefff & Posech | |

| | | | | | |
|---|---|---|---|---|---|
| | Nombre mois | CCP préced Du - Au | CCP préced Du - Au | CCP préced Du - Au | Total mois |
| | | Signaletic. Cdsd & Cdsf | Signaletic. Cdsd & Cdsf | Signaletic. Cdsd & Cdsf | |
| | | Fg-Pers. Poseff & Posech | Fg-Pers. Poseff & Posech | Fg-Pers. Poseff & Posech | |

*C) Interactions avec l'historique.*

Oui. Le programme devra aller chercher pour chaque fonctionnaire en CCP ses trois derniers CCP (date de début & date de fin), lesquels se trouvent dans le fichier historique.

*D) Des modifications a effectuer.*
Non, la sous-fonctionnalité ne nécessite pas de faire des mises-à-jour.

Par conséquent le programme doit être conçu:

-soit au moyen de Natural

-soit au moyen de la fonction "Exploitation de l'historique"

*E) Si on veut le réaliser au moyen du langage de commande Natural*

Comme on a pu le constater au cours du point 5 de la Partie 1 (traitement de l'historique), le problème d'historisation des données n'est pas simple. Il en est de même lorsqu'on veut récupérer des valeurs de données historisées dans le fichier historique.
Dans ce qui suit, nous allons donner des approfondissements du point 5 de la Partie 1 de manière à mieux cerner le problème.

La consultation de données au sein d'Adabas s'effectue par l'intermédiaire d'index. Ainsi, pour chaque fichier de la BD Arpege, il existe (au moins) un fichier "index" permettant à l'utilisateur d'accéder à des enregistrements en fonction de la valeur d'une zone (un descripteur)

Sous Adabas un descripteur peut se présenter sous trois formes différentes:

- *Descripteur normal*: index sur une zone

-*Sous-descripteur*: c'est un descripteur basé sur une partie d'une zone : Par exemple, si la zone = Nom, un sous-descripteur a la forme suivante:

*Nom, La, Lb, val*

où *Nom* est la zone concernée, *La* est la position du caractère par où on va commencer, *Lb* est la position du caractère de fin , et *val* est la valeur de la chaîne de caractères comprise entre La et Lb. Ainsi pour la recherche de toutes les personnes ayant un nom commençant par ABC on aura comme descripteur Nom,1, 4, ABC.

P.S. Le descripteur normal est un sous-descripteur où Lb - La = longueur de la zone.

-Super-descripteur: c'est la conjonction de sous-descripteurs:

zone 1, La1, Lb1, Val1
et zone 2, La2, Lb2, Val2
et zone 3, La3, Lb3, Val3

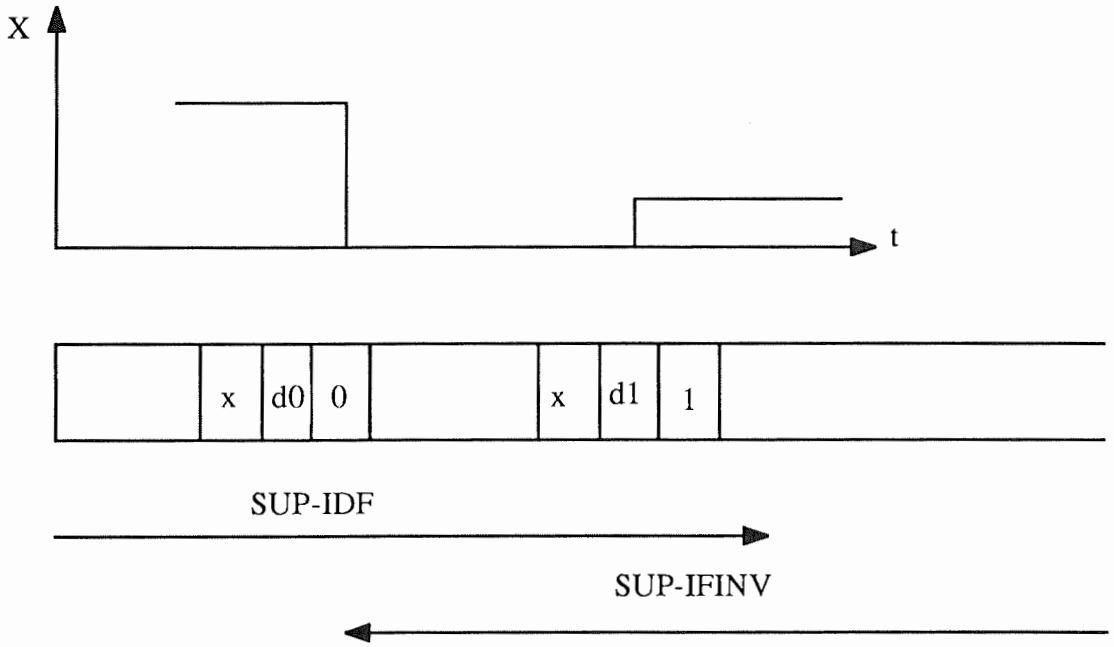
La gestion de l'historique est basée sur les super-descripteurs. En effet, comme l'identifiant du fichier historique est :

-rec-id
-filename
-fieldname
-date
-valeur (séquence)

Il se trouve que toute recherche dans ce fichier doit s'effectuer avec un Super-descripteur composé des mêmes éléments que cet identifiant. Il existe dans Arpege deux Super-descripteurs utilisés pour la recherche dans l'historique:

*-le Super-descripteur Sup: SUP-IDF*

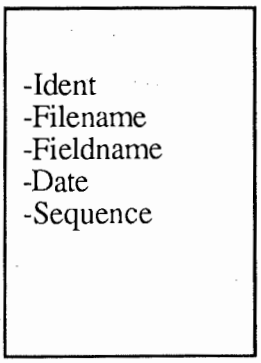
Il parcours le fichier historique par date chronologique (de la gauche vers la droite)



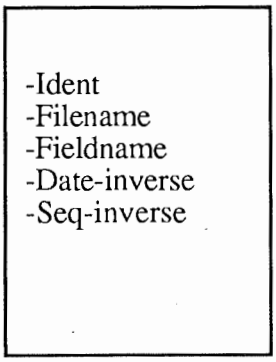

*-le Super-descripteur inverse: SUP-IFINV*

Il parcours le fichier historique par date anti-chronologique (remonte le fichier de la droite vers la gauche)

SUPER DESCRIPTEUR




SUPER DESCRIPTEUR INVERSE




Exemple Date :
d0 = 1/1/82 ( 9820101)
d1 = 1/4/82 (19820401)

d1 > d0

==>

Exemple Date-inverse:
d0 = 99999999-19820101 = 80179898
d1 = 99999999-19820401 = 80179598

d0 > d1

L'application Chronos comporte un ensemble de modules (qui interagissent entre eux) rendant des services de consultation au programmeur. Ainsi, le programme atomique (il n'appelle aucun autre programme) *N#STORY* recherche la valeur d'une zone à une date donnée.

Voir programme.

```
      ************************************************************************
0020  *                                                                      *
0030  * NOM : N#HSTORY     DATE ECRIT. :   1/08/88      AUTEUR : F. GABARRON  *
0040  * ===                                                                  *
0050  * VERSION : 1.2      DATE MODIF. :  15/02/89      AUTEUR : D. DECHANET  *
0060  * ==========                                                           *
0070  * OBJET MODIF. :                                                       *
0080  * ==============                                                       *
0090  *    ADAPTATION MODELE W.NIJMAN AUX BESOINS DE ARPEGE !!!               *
0100  *    AJOUT PREFIXE 'FILENAME'.                                         *
0110  *    AJOUT DE 4 CHAMPS EN OUTPUT                                       *
0120  *                                                                      *
0130  *                                                                      *
0140  ************************************************************************
0150  *                                                                      *
0160  * FONCTION :  RECHERCHE LA VALEUR D'UN FIELD A UNE DATE DONNEE...      *
0170  * ==========                                                           *
0180  *                                                                      *
0190  * LIENS HIERARCHIQUES :                                                *
0200  * ======================                                               *
      *   CALLED BY       : P#16. P#17. P#26. P#36. N#MAJPDF. R#CONSF1.      *
0220  *                     R#CONSE1. N#EDETST. N#VCEMP. N#VWEMP. N#TSTDEL.  *
0230  *                     R#CONSPX (X=1..5)                                *
0240  *   CALLING         :                                                  *
0250  *   INTERNAL CALLS  :                                                  *
0260  *   MAPS APPELES    :                                                  *
0270  *                                                                      *
0280  ************************************************************************
0290  *
0300  DEFINE DATA PARAMETER
0310  *           ***********
0320  1 #FILENAME(A1)                        /* INPUT : IDENTIFIES MASTER FILE
0330  1 #IDENT (B4)                          /* INPUT : IDENTIFIES MASTER RECORD
0340  1 #DATE  (N8)                          /* INPUT : SPECIFIES DATE
0350  1 #FIELD (A32)                         /* INPUT : SPECIFIES FIELD
0360  1 #VALUE (A253)                        /* OUTPUT : VALUE OF FIELD
0370  1 #DATER (N8)                          /* OUTPUT : REAL DATE FOUND
0380  1 #CAUSE (N3)                          /* OUTPUT : MODCOD
0390  1 #DATED (N8)                          /* OUTPUT : DATE DECISION
0400  1 #DATEM (N8)                          /* OUTPUT : DATE MODIFICATION
      1 #MULTI (L)                           /* OUTPUT : INDICATES MULTIPLE HIST.
0420  *
0430              LOCAL
0440  *           *******
0450  1 HIST VIEW OF FG-HISTORY              /* View of History File
0460    2 FILENAME
0470    2 IDENT
0480    2 FIELDNAME
0490    2 FIELDVALUE
0500    2 INV-DATE
0510    2 DATE
0520    2 MODCOD
0530    2 DATDEC
0540    2 DATMAJ
0550  *
0560  1 #SUPVAL       (A26)                  /* STRUCTURE OF SUPERDESCRIPTOR
0570  1 REDEFINE #SUPVAL
0580    2 #SUP-FILENAME(A1)
0590    2 #SUP-IDENT (B4)
```

```
0620 *
0630 1 #SAVDAT (N8)                          /* To detect multiple history recs
0640 *
0650 END-DEFINE
0660 *
0670 *
0680 RESET #VALUE #SAVDAT #CAUSE #DATER #DATEM #DATED /* INIT RESULT FIELDS
0690 MOVE FALSE TO #MULTI
0700 *
0710 MOVE #FIELD TO #SUP-FIELD              /* Construct superdescriptor
0720 MOVE #IDENT TO #SUP-IDENT
0730 MOVE #FILENAME TO #SUP-FILENAME
0740 COMPUTE #SUP-DATE = 99999999 - #DATE
0750 *
0760 *      Note : SUP-IFINV is on IDENT. FIELDNAME and INVERTED DATE
0770 *
0780 READ HIST LOGICAL BY SUP-IFINV STARTING FROM #SUPVAL
0790 *
0800 *      The following IF is necessary because an ENDING AT clause is
     *      not allowed with superdescriptors
0820 *
0830   IF HIST.IDENT NE #SUP-IDENT  OR
0840      HIST.FILENAME  NE #SUP-FILENAME OR
0850      HIST.FIELDNAME NE #SUP-FIELD
0860     ESCAPE ROUTINE
0870   END-IF
0880 *
0890   IF #SAVDAT = 0                                  /* FIRST VALUE FOUND
0900 *                                                 /* (i.e. most recent)
0910     MOVE HIST.FIELDVALUE TO #VALUE
0920     MOVE HIST.INV-DATE   TO #SAVDAT              /* SAVE DATE FOR CHECK
0930     MOVE HIST.DATE TO #DATER
0940     MOVE HIST.MODCOD TO #CAUSE
0950     MOVE HIST.DATDEC TO #DATED
0960     MOVE HIST.DATMAJ TO #DATEM
0970 *
0980   ELSE
0990 *
1000     IF HIST.INV-DATE = #SAVDAT                  /* SECOND VALUE FOUND
           MOVE TRUE TO #MULTI                       /* Indicate multiple if
1020     END-IF                                      /* DATE IS THE SAME
1030 *
1040     ESCAPE ROUTINE
1050 *
1060   END-IF
1070 *
1080 END-READ
1090 *
1100 END
```

A la ligne 780, on peut remarquer l'instruction:

READ HIST LOGICAL BY SUP-IFINV STARTING FROM #SUPVAL

Elle consiste à effectuer une recherche "en arrière" au moyen du Super-descripteur inverse. Ce programme recherche donc la valeur d'une zone à une date donnée.

*Comment reconstituer un enregistrement d'un des fichiers (personnel, famille, emplois) tout entier ?*

Il suffira d'appeler le programme *N#STORY* autant de fois qu'il y a de champs historisés. C'est ainsi que le programme *R#CONSP4* de la librairie Chronos reconstitue plusieurs zones d'un record.

Voir programme.

```
0040 *                                                                      *
0050 * Nom : P#CONSP4      DATE FINALE : 27/02/89      AUTEUR : F. GABARRON  *
0060 * ===                                                                  *
0070 * VERSION : 1.3       DATE MODIF. : 27/02/89      AUTEUR : F. GABARRON  *
0080 * ==========                                                           *
0090 * OBJET MODIF. :                                                       *
0100 * ==============                                                       *
0110 *      AJOUT PREFIXE 'FILENAME' DANS HISTORIQUE.                       *
0120 *      MODIF TRAITEMENT DATE RECONSTITUTION.                           *
0130 *      MODIF GESTION MAJEFF (RECUPERER MODCOD ET DATDEC)               *
0140 *                                                                      *
0150 ************************************************************************
0160 *                                                                      *
0170 * FONCTION :  CONSTRUCTION RECORD PERSONNEL A UNE DATE DONNEE...       *
0180 * ==========    (PARTIE CORRESPONDANTE A L'ECRAN S#PERS4!!!)           *
0190 *                                                                      *
0200 * LIENS HIERARCHIQUES :                                               *
0210 * ======================                                              *
     *    CALLED BY        : P#14, P#17                                    *
0220 *                                                                      *
0230 *    CALLING          : N#DATIN, N#HSTORY                             *
0240 *                                                                      *
0250 *    INTERNAL CALLS   : CALL-HISTORY                                  *
0260 *                                                                      *
0270 *    MAPS APPELES     : W#HISWAR                                      *
0280 *                                                                      *
0290 ************************************************************************
0300 *
0310 DEFINE DATA GLOBAL USING GIANT-GL WITH GIANT-BLOCK.HISTO-BLOCK
0320 *           ********                    ************************
0330              LOCAL USING PERS-LDA
0340 *           *******     **********
0350              LOCAL
0360 *           *******
0370 1 #FIELDNAME     (A32)                 /* Workfields for communication
0380 1 #VALUE         (A253)
0390 1 #DATREL        (N8)
0400 1 #MODCOD        (N3)
     1 #DATDEC        (N8)
0420 1 #DATMAJ        (N8)
0430 1 #FILENAME      (A1)   INIT <'P'>
0440 1 #MULTIPLE      (L)
0450 *
0460 1 HIST VIEW OF FG-HISTORY              /* View of History File
0470   2 FILENAME                           /* Identifies Master File
0480   2 IDENT                              /* Used to verify required date
0490   2 DATE
0500 *
0510 1 #SUPER         (A26)                 /* Structure of superdescriptor
0520 1 REDEFINE #SUPER
0530   2 #SUP-FILENAME(A1)
0540   2 #SUP-IDENT   (B4)
0550   2 #SUP-DATE    (N8)
0560 *
0570 1 #WORK-CV       (C)                   /* Value for CV
0580 1 REDEFINE #WORK-CV
0590   2 #WCV         (B2)
```

```
0640 END-DEFINE
0650 *
0660 DEFINE SUBROUTINE R#CONSP4
0670 ****************************
0680 *
0690 MOVE FG-PERS-VIEW.IDENT TO #SUP-IDENT     /* Construct Superdescriptor
0700 MOVE #FILENAME TO #SUP-FILENAME
0710 RESET #SUP-DATE
0720 *
0730 *              Note : SUP-IDF is on IDENT and DATE
0740 *
0750 READ (1) HIST LOGICAL BY SUP-IDF STARTING FROM #SUPER
0760 *
0770 IF HIST.IDENT NE #SUP-IDENT OR        /* No history at all
0780    HIST.FILENAME NE #SUP-FILENAME
0790    ESCAPE ROUTINE
0800    END-IF
     *
0820 IF #CHRONOS-DATE < HIST.DATE          /* Asked date is too old. use S.I.
0830    IF #CHRONOS-MODE LT 7
0840       MOVE HIST.DATE TO #CHRONOS-DATE  /* Take oldest date
0850    ELSE
0860       IF +TRX-PG-ACTIF = ' '           /* Warning pas encore affiche
0870          CALLNAT 'N#DATIN' #CHRONOS-DATE #DATE-DEM
0880          CALLNAT 'N#DATIN' #CHRONOS-DATE #DATE-MIN
0890          INPUT USING MAP 'W#HISWAR'    /* Affichage Warning
0900          MOVE '0' TO +TRX-PG-ACTIF
0910       END-IF
0920    END-IF
0930    ESCAPE ROUTINE
0940 END-IF
0950 *
0960 END-READ
0970 *
0980 MOVE #CV-TABLE (*) TO #CV-TAB (*)
0990 *
1000 MOVE 'PRSEFF'  TO #FIELDNAME
     MOVE #CV-PRSEFF TO #WORK-CV
1020 PERFORM CALL-HISTORY
1030 IF #VALUE NE ' '
1040    COMPUTE FG-PERS-VIEW.PRSEFF = VAL (#VALUE)
1050    ELSE
1060       RESET FG-PERS-VIEW.PRSEFF
1070    END-IF
1080 MOVE #WORK-CV   TO #CV-PRSEFF
1090 *
1100 MOVE 'PRSTYPC'  TO #FIELDNAME
1110 MOVE #CV-PRSTYPC TO #WORK-CV
1120 PERFORM CALL-HISTORY
1130 MOVE #VALUE     TO FG-PERS-VIEW.PRSTYPC
1140 MOVE #WORK-CV   TO #CV-PRSTYPC
1150 *
1160 MOVE 'PRSTYPL'  TO #FIELDNAME
1170 MOVE #CV-PRSTYPL TO #WORK-CV
1180 PERFORM CALL-HISTORY
1190 MOVE #VALUE     TO FG-PERS-VIEW.PRSTYPL
1200 MOVE #WORK-CV   TO #CV-PRSTYPL
     *
2860 *
2870 *
2880 DEFINE SUBROUTINE CALL-HISTORY
2890 *                 **************
2900 CALLNAT 'N#HSTORY' #FILENAME          /* Identifies Master File
2910                    FG-PERS-VIEW.IDENT /* Internal Ident from Record
2920                    #CHRONOS-DATE      /* Date
2930                    #FIELDNAME         /* Name of field
2940                    #VALUE             /* Will contain result value
2950                    #DATREL            /* Will contain real histo-date
2960                    #MODCOD            /* Will contain modif-code
```

```
3000 *
3010 IF #MULTIPLE                              /* Determine value of CV
3020    DECIDE ON FIRST VALUE OF #WCV          /* It should blink when multiple
3030       VALUES H'F800'.H'5800'              /* Prot. et IP
                  MOVE (AD=BP) TO #WORK-CV
3050       VALUES H'F000'.H'5000'              /* Disp. et Intens.
3060              MOVE (AD=B)  TO #WORK-CV
3070       NONE VALUE IGNORE                   /* Invisible reste inchange
3080    END-DECIDE
3090 END-IF
3100 *
3110 END-SUBROUTINE                            /* call-history
3120 *
3130 END-SUBROUTINE                            /* r#conso4
3140 *
3150 END
```

On peut voir que de la ligne 980 à la ligne 1200 le programme ne fait que reconstituer les champs recherchés avec à chaque fois (via la subroutine Call-history) appel au programme atomique N#STORY.

On se rend compte que la réalisation de notre sous-fonctionnalité suppose une bonne connaissance des programme de l'application Chronos. La complexité de la gestion de cet historique rend la tâche assez complexe. C'est pour cette raison que le P.E. a développé une fonction "Exploitation de l'historique" dont nous allons nous servir pour résoudre notre problème.

*6) Fonction Exploitation de l'historique*

Comme déjà vu, cette fonction comporte trois phases:

1) sélection de la population de base
2) Optionnelle: sélection supplémentaire. Epuration de la population de base par application d'une équation sur l'historique.
3) Reconstitution des données pour cette population:
   -pour certains champs demandés
   -à une date donnée, pour une période donnée (les dates peuvent être dynamiques)

Voir les écrans de la fonction pages suivantes.

**Comment pourrait-on résoudre notre cas ?**

**1°.**Obtention de toutes les informations (de l'historique) pour tous les fonctionnaires intéressés (via la fonction exploitation historique)

**2°.**Les informations seront traitées avec Super Natural de manière à effectuer les calculs voulus et à fournir la présentation désirée.

*1°. Fonction Exploitation de l'historique.*

1) sélection population :position administrative (POSTYPC) = '30' c'est-à-dire qui sont en CCP.
2) période à reconstituer : ENTPAR --> POSEFF

3) Mouvements à reconstituer: MAJMODC (cause de modification) = 12, 13, 15, 16, 21

12: réintégration pendant un CCP
13: réintégration à l'issue d'un CCP
15: octroi d'un CCP
16: prolongation d'un CCP
21: instance reintégration issue d'un CCP

On ne se préoccupe donc, que pour les fois où la cause de modification concerne le CCP.

4) Données à reconstituer : NOM/ POSTYPC/ POSEFF/ POSECH/ CLSCAT/ CLSGRA/ ETCNAT
(ce sont les champs qui nous intéressent)

5) Critère de tri: NOPERS/ MAJEFF (descending)

*2°. Super Natural*

Exploration des records reconstitués par l'étape précédente et édition de la liste suivant le format.
Le seul problème est que la fonction Exploitation historique sélectionne tous les CCP des fonctionnaires car on ne peut limiter la recherche. Par conséquent on obtiendra une solution telle que la suivante:

| | | | | CCP | | |
|---|---|---|---|---|---|---|
| Nom | Prénom | Classement | Nationalité | DU | AU | Total mois |
| XXX | ... | .... | ..... | xx | xx | xxx |
| | | | | xx | xx | xxx |
| | | | | xx | xx | xxx |
| | | | | | | ---------- |
| | | | | | | xxxxxx |
| YYY | ... | ... | ... | yy | yy | yyy |
| | | | | yy | yy | yyy |

16/04/91 16:03:06

Exploitation Historique

Selection des causes modification a reconstituer

| CHOIX | CODE | LIBELLE |
|---|---|---|
| | 001 | PROMOTION (ART. 45 AL. 1) |
| | 002 | NOM. FONCT. STAG. CONCOURS INTERNE |
| | 003 | NOM. AUTRE CAR. SUITE CONCOURS EXT |
| | 004 | NOM. AUTRE CAR. SUITE CONCOURS INT |
| | 005 | PROMOTION AUTRE CARRIERE |
| | 006 | RECLASSEMENT |
| | 007 | CHANGEMENT DE FONCTIONS |
| | 008 | NOMIN. FONCT. STAG. CONCOURS GENE. |
| | 009 | PROCEDURE SPECIALE DE RECRUTEMENT |
| x | 010 | AVANCEMENT D'ECHELON |
| | 011 | DECISION DU BUREAU |
| | 012 | REINTEGRATION PENDANT UN C.C.P. |

F1=RECOMMENCER F16=PAGE+1 F20=RETOUR/MENU

VEUILLEZ CONFIRMER VOTRE CHOIX :

1. CONFIRMATION DEMANDE HISTORIQUE
2. RECOMMENCER CHOIX CAUSES MODIF.

9. ABANDON COMPLET DEMANDE HISTORIQUE

1

SAISIR UN NUMERO

```
+ ------------------------------------------------------------------------ +
I                      LISTE DES SUBFILES DISPONIBLES                      I
I  GABE                      POUR  CE  USER                      16/04/91 I
+ ------------------------------------------------------------------------ +


                   Vous disposez de  : 03  Subfiles
              15   14   13

                Les numeros suivants sont deja utilises
              15   14   13

                Saisir le numero du subfile sur lequel
                vous desirez travailler  : --

                Ainsi que le nombre de jour pendant lequel vous
                voulez le garder         : --   ('10' par defaut)

                Si vous voulez faire de la place et vider un subfile
                saisir son numero        : --

                        ( PF20 = Abandon)



+ ------------------------------------------------------------------------ +
I                     EXPLOITATION DE L'HISTORIQUE                         I
I  GABE                     MENU DE SELECTION                     16/04/91 I
+ ------------------------------------------------------------------------ +


                  Choisir le ou les fichiers a partir
              desquels vous voulez effectuer une selection


                        1 : Personnel
                        2 : Poste
                        3 : Personnel/Poste
                        4 : Poste/Personnel
                        5 : Personnel/famille


                            Choix : -


                     ( PF20 = Quitter la fonction )
```

```
+ ------------------------------------------------------------------------ +
I                      SELECTION DES ZONES QUI SERONT                      I
I  GABE                 CONSERVEES POUR EXPLOITATION              16/04/91 I
+ ------------------------------------------------------------------------ +
   FICHIER : PERSONNEL
   Marquer les zones choisies
 _ HSTEFF          _ LNGETR5         _ ACCADR1         _ ETCNAT
 x NOPERS          _ LNGCOR          _ ACCADR2         _ ETCNAT2
 _ NOPERSS         _ ADRADM          _ ACCTEL          x AFFVILC
 _ ETCTIT          _ ADRADMS         _ MATR            x AFFVILL
 _ ETCNOM          _ ADRTEL          _ MAJDECB         x AFFPAY
 _ ETCDNA          _ HABPAY          _ AUXOBS          x AFFMOD
 _ ETCPNA          _ HABVILL         _ NOAVIS          x AFFEFF
 _ ETCLNA          _ HABZIP          _ AUXDUR          _ AFFVILCB
 _ ETCNNA          _ HABADR1         _ LSTMOD          _ AFFVILLB
 _ RECNAT          _ HABADR2         _ MAJNUM          _ AFFPAYB
 _ LNGPRI          _ HABTEL          _ NOTECH          _ AFFMODB
 _ LNGETR1         _ ACCTIT          _ RISEFF          _ AFFEFFB
 _ LNGETR2         _ ACCNOM          _ AFFINS          _ AUXTYP
 _ LNGETR3         _ ACCPAY          x NOM             _ LSTTYPC
 _ LNGETR4         _ ACCVILL         x PRENOM          _ LSTTYPL
      ( ENTER = pagination      PF19 = Execution      PF20 = Abandon)
```

```
DESIREZ VOUS RECONSTITUER :
===========================
      TOUS LES MOUVEMENTS (T)    OU
      CERTAINES CAUSES MODIF. SEULEMENT (C) c
```

```
+ ------------------------------------------------------------------------ +
I                      SELECTION DES CRITERES DE RECHERCHE                 I
I  GABE                                                          16/04/91 I
+ ------------------------------------------------------------------------ +
 FICHIER
 PERSONNEL
        ZONE                          EQUATIONS                    OPERATEUR
   nopers---------- ) '015000'---------------------------------- and--
   nopers---------- ( '017000'---------------------------------- -----
   ---------------- ---------------------------------------------- -----
   ---------------- ---------------------------------------------- -----
   ---------------- ---------------------------------------------- -----
   ---------------- ---------------------------------------------- -----
   ---------------- ---------------------------------------------- -----
   ---------------- ---------------------------------------------- -----
   ---------------- ---------------------------------------------- -----
   ---------------- ---------------------------------------------- -----
 Saisir les criteres de tri
   nom/prenom/nopers-------------------------------------------------------

       ( PF18 = Selection complementaire sur l'historique
         PF19 = Executer la fonction    PF20 = Quitter la fonction )



+ ------------------------------------------------------------------------ +
I                       EXPLOITATION DE L'HISTORIQUE                       I
I  GABE                  SELECTION DE LA SITUATION               16/04/91 I
+ ------------------------------------------------------------------------ +


      -   Si vous voulez la situation INITIALE   : -   (X)


      -   Si vous desirez une reconstitution

            A UNE DATE DONNEE, saisir la date   : 01/01/90
            ENTRE DEUX DATES, saisir les deux   : 31/12/90


      -   Vous pouvez demander une reconstitution sur la base d'une
          DATE SE TROUVANT DANS LA SITUATION FINALE

            POUR UNE DATE, saisir la zone       : ----------
            ENTRE DEUX DATES, saisir les deux   : ----------

      ( PF19 = Executer la fonction    PF20 = Quitter la fonction )
```

Bien qu'on s'écarte un peu de la présentation attendue, la solution obtenue offre plus d'informations concernant les CCP d'un fonctionnaire.

### 4.5. Enfants changeant de code allocation scolaire.

*A) Enoncé:* Liste des fonctionnaires ayant de enfants qui vont changer automatiquement de code allocation scolaire (quand ils vont avoir l'âge de 11 ans, c'est-à dire quand ils vont passer des études primaires aux études secondaires)

*B) Identification des champs:*

| Appellation | Matricule fonct. | Nom fonct. | Prenom fonct. | Prénom enfant | Date de naissance |
|---|---|---|---|---|---|
| Dans Isdam | Signaletic. Nrmatr | Signaletic. Nom | Signaletic. Prenom | Signaletic. Epre | Signaletic. Edna |
| Dans Arpege | Fg-Pers. Nopers | Fg-Pers. Nom | Fg-Pers. Prenom | Fg-Fami. Pacprn | Fg-Fami. Pacdna |

*C) Interactions avec l'historique.*

Aucune nécessité de consultation de l'historique.

*D) Mises-à-jour à effectuer.*

La sous-fonctionnalité aura à effectuer des modifications dans la BD Arpege. Ces modifications concernent le changement du code allocation scolaire.(SCOTYP):

(SCOTYP<-- + 11)

Conséquences de C) et D) : la sous-fonctionnalité devra être effectuée

- au moyen de la fonction SELECT
- avec Natural.

On pourrait ici utiliser la fonction SELECT, car le nombre d'enfants concernés est assez petit.

Mais, étant donné qu'il s'agisse d'une procédure à effectuer périodiquement (tous les mois), il serait préférable de concevoir un programme Natural.

*E) Le programme Natural*

Le programme Natural va se décomposer en deux parties:

- la première partie *CM-AL-TP* : qui se chargera de faire l'interface "on-line" avec l'utilisateur. Il lancera, via le Launcher, le programme batch *CM-AL-BT*. Il se situera dans les modules de la partie on-line de l'application Arpege.
- la seconde partie *CM-AL-BT* : programme batch s'occupant de répondre aux besoins de la sous-fonctionnalité. C'est la partie qui réalise la sélection et la modification dans la BD Arpege.

F) *CM-AL-TP* : de manière similaire à ce qu'il a été fait dans 4.3.

Voir programme.

G) *CM-AL-BT*

Le critère de sélection :

-fonctionnaires ayant la position administrative POSTYPC < 24 (on ne prend pas ceux qui sont en CCP ou qui sont détachées, ...)

-fonctionnaires ayant des enfants qui vont atteindre l'âge de onze ans.

Le programme Natural CM-AL-BT:

```
NEXT LIST CM-AL-TP
0010 ************************************************************************
0020 *                                                                      *
0030 * NOM : CM-AL-TP                                                       *
0040 * ===                                                                  *
0050 *                                                                      *
0060 * OBJET                                                                *
0070 * =====                                                                *
0080 *        LANCEMENT BATCH GESTION DES CHANGEMENT CODE ALLOC. SCO.        *
0090 *                                                                      *
0100 ************************************************************************
0110 *                                                                      *
0120 * FONCTION :  PROGRAMME DE LANCEMENT EN BATCH DES FONCTIONNAIRES       *
0130 * ==========  AYANT DES ENFANTS QUI VONT CHANGER DE CODE ALLOC. SCOL. * *
0140 *                                                                      *
0150 * LIENS HIERARCHIQUES :                                                *
0160 * =====================                                                *
0170 *       PGM.APPELANT : MDRIVER (VIA COMMANDES)                         *
0180 *     SOUS-FONCTIONS: LAUNCHER                                         *
0190 *                                                                      *
0200 *       MAPS APPELES : /                                               *
0210 *                                                                      *
0220 ************************************************************************
0230 *
0240 DEFINE DATA LOCAL
0250 * ***************
0260 *
0270 1 RJE-PARAMETERS
0280   2 RJE-REQUEST-NUMBER          (A4)
0290   2 RJE-TSN-NUMBER              (N4)
0300   2 RJE-TIME-LIMIT              (N4)
0310   2 RJE-DATABASE-ID             (N1)
0320   2 RJE-APPLICATION-ID          (A8)
0330   2 RJE-PROGRAM                 (A8)
0340   2 RJE-USER-ID                 (A8)
0350   2 RJE-USER-PASSWORD           (A8)
0360   2 RJE-FORM-TYPE               (A4)
0370   2 RJE-DESTINATION             (A4)
0380   2 RJE-IMMEDIATE-ENTER         (A1)
0390   2 RJE-SYSLST                  (A1)
0400   2 RJE-NUMBER-OF-COPIES        (N1)
0410   2 RJE-NUMBER-OF-WORKFILES     (N1)
0420   2 RJE-NUMBER-OF-PRINTFILES    (N1)
0430   2 RJE-DATA                    (A80)
0440   2 RJE-RETURN-CODE             (N2)
0450 *
0460 1 #ERR-TEXT(A60)
0470 1 #ETAT     (A30)
0480 1 #NCOPIES (N1)
0490 *
0500 1 #DATA     (A80)
0510 1 REDEFINE #DATA
0520   2 #CHOIX (A1)
0530   2 #FILLER(A79)
0540 *
0550 END-DEFINE
0560 *
0570 * ---------------------------------------------------------------------
0580 *
0590 RELEASE STACK
```

```
0600 RESET #DATA
0610 *
0620 INPUT  10X 'MENU SELECTION' /
0630        10X '-' (17) //
0640        10X '1. SIMPLE LISTE' /
0650        10X '2. MISE A JOUR SANS LISTE' /
0660        10X '3. MISE A JOUR AVEC LISTE' /
0670        10X ' . QUITTER' ///
0680        10X 'ENTREZ LA SELECTION :' #CHOIX
0690 IF #CHOIX = '1' OR = '2' OR = '3' OR = ' '
0700    IGNORE      /* O.K.
0710 ELSE
0720    REINPUT 'CHOIX ERRONE !!!'
0730 END-IF
0740 *
0750 *
0760 *  LANCEMENT BATCH...
0770 *
0780 MOVE 'CM-AL-BT' TO RJE-PROGRAM
0790 MOVE 1          TO RJE-NUMBER-OF-PRINTFILES
0800 MOVE 60         TO RJE-TIME-LIMIT
0810 MOVE 'Y'        TO RJE-IMMEDIATE-ENTER
0820 MOVE 'Y'        TO RJE-SYSLST
0830 MOVE #DATA      TO RJE-DATA
0840 MOVE 'SITE'     TO RJE-DESTINATION                    /* 'LBAK' = LASER
0850 * MOVE 'LASA'   TO RJE-FORM-TYPE                      /* FORMAT LASER
0860 *
0870 CALLNAT 'LAUNCHER' RJE-PARAMETERS
0880 *
0890 IF RJE-RETURN-CODE NE 0
0900    INPUT USING MAP 'H#EBATCH'
0910    FETCH 'MDRIVER' #ERR-TEXT
0920 ELSE
0930    COMPRESS 'LST.PROD.LIST' RJE-REQUEST-NUMBER INTO #ETAT LEAVING NO
0940    SET KEY PF20
0950    SET CONTROL 'WFL53C10B5/14<<--'
0960    INPUT USING MAP 'H#GLOSER'
0970    SET CONTROL 'WML79C23B<<--'
0980    FETCH 'MDRIVER' '0131'
0990 END-IF
1000 *
1010 *
1020 *
1030 END
***** End of List *****
```

```
NEXT LIST CM-AL-BT
0010 ************************************************************************
0020 * PROGRAMME: CM-ALLOCBT                                                *
0030 * OBJET : EFFECTUE LA PARTIE BATCH DU PROGRAMME AYANT                  *
0040 *         COMME BUT DE GERER LES FONCTIONNAIRES QUI ONT                *
0050 *         DES ENFANTS CHANGEANT DE CODE ALLOCATION SCOLAIRE  *
0060 * DATE ECR. 10-05-91                                                   *
0070 * AUTEUR: C. SAITTA                                                    *
0080 *                                                                      *
0090 * IL APPELLE :  N#HSTUPD                                               *
0100 *                                                                      *
0110 * APPELLE PAR : VIA LE LAUNCHER PAR CM-ALLOCTB                         *
0120 *                                                                      *
0130 ************************************************************************
0140 *
0150 DEFINE DATA LOCAL
0160 *
0170 1 PERSONNEL-VIEW VIEW OF FG-PERS
0180   2 NOPERS
0190   2 NOM
0200   2 PRENOM
0210   2 LSTTYPC
0220   2 POSTYPC
0230   2 ENFNBR
0240 ****** POUR L'HISTORISATION ... ************
0250   2 IDENT
0260   2 MAJEFF
0270   2 MAJMODC
0280   2 MAJDEC
0290 1 PAC-VIEW VIEW OF FG-FAMI
0300   2 PACREF
0310   2 PACPRN
0320   2 PACDNA
0330   2 SCOTYP
0340   2 PACTYP
0350 *
0360 1 #PARAM (A80)
0370 *
0380 1 REDEFINE #PARAM
0390   2 #CHOIX (A1)
0400   2 #FILLER(A79)
0410 *
0420 1 #DATE-11ANS (N8)
0430 1 #DATE-CALC (N8)
0440 1 #DATE-DEB (N8)
0450 1 REDEFINE #DATE-DEB
0460   2 #DD-AAMM(N6)
0470   2 #DD-JJ   (N2)
0480 1 #DATE-FIN (N8)
0490 1 ##NUM-ERR (N4)
0500 *
0510 *******     PARAMETRES POUR L'HISTORISATION ...
0520 *
0530   1 #FILENAME    (A1) INIT <'P'>
0540   1 #IDENT       (B4)
0550   1 #DATE        (N8)
0560   1 #FIELDNAME   (A32)
0570   1 #FIELDVALUE  (A50)
0580   1 #MODCOD      (N3)
0590   1 #DATDEC      (N8)
```

```
0600    1 #USER-OPER   (A6)
0610    1 #RESULT      (L)
0620    1 #MAJ-SF      (L)
0630 *
0640    1 HIST VIEW OF FG-HISTORY
0650      2 FIELDNAME
0660      2 DATE
0670      2 MODCOD
0680      2 DATDEC
0690    1 #SUP-IDF     (A26)
0700    1 REDEFINE #SUP-IDF
0710      2 #SFILE     (A1)
0720      2 #SIDENT    (B4)
0730      2 #SDATE     (N8)
0740      2 #SFIELD    (A10)
0750 ********** POUR LA GESTION COHERENTE DE L'HISTORIQUE ...
0760    1 #W-DATE(A8)
0770    1 #SW-WARNERR(L)
0780 END-DEFINE
0790 *
0800 ****** FIN DE DECLARATION DES DONNEES ****************************
0810 *
0820 FORMAT LS=130
0830 FORMAT (1) PS=55 LS=132 IS=ON
0840 * ************************************************************* '
0850 INPUT #PARAM
0860 * ************************************************************
0870 MOVE *DATN TO #DATE-DEB                    /* DEBUT SELECTION MOIS
0880 MOVE 1     TO #DD-JJ
0890 COMPUTE #DATE-FIN   = #DATE-DEB + 30       /* FIN SELECT. MOIS (+30 JOURS)
0900 COMPUTE #DATE-11ANS = PAC-VIEW.PACDNA + 110000   /* +11 ANS
0910 *
0920 * ************************************************************
0930 DECIDE ON FIRST VALUE OF #CHOIX
0940        VALUE '1'    PERFORM LISTE
0950        VALUE '2'    PERFORM MAJ
0960        VALUE '3'    PERFORM MAJ-LISTE
0970        NONE VALUE   TERMINATE
0980 END-DECIDE
0990 *
1000 ********** LES TROIS ROUTINES ...*****************************
1010 *** 1. SI ON NE DEMANDE QU'UNE LISTE...
1020 *
1030 DEFINE SUBROUTINE LISTE
1040 *
1050 READ PERSONNEL-VIEW BY NOM
1060  ACCEPT IF PERSONNEL-VIEW.LSTTYPC < 7 AND PERSONNEL-VIEW.POSTYPC < 31
1070              AND PERSONNEL-VIEW.ENFNBR > 0
1080  FIND PAC-VIEW WITH PACREF = NOPERS
1090     IF PACTYP = 'ENF' AND
1100                (#DATE-11ANS GE #DATE-DEB AND  #DATE-11ANS LE #DATE-FIN)
1110        DISPLAY NOTITLE PERSONNEL-VIEW.NOM PERSONNEL-VIEW.PRENOM
1120                  PAC-VIEW.PACPRN PAC-VIEW.PACDNA PAC-VIEW.PACREF
1130     END-IF
1140  END-FIND
1150 END-READ
1160 END-SUBROUTINE
1170 *
1180 ************************************************************
1190 ** 2. SI ON DEMANDE UNE SIMPLE MISE-A-JOUR....
```

```
1200 *
1210 DEFINE SUBROUTINE MAJ
1220 *
1230 READ PERSONNEL-VIEW BY NOM
1240  ACCEPT IF PERSONNEL-VIEW.LSTTYPC < 7 AND PERSONNEL-VIEW.POSTYPC < 31
1250             AND PERSONNEL-VIEW.ENFNBR > 0
1260  FIND PAC-VIEW WITH PACREF = NOPERS
1270     IF PACTYP = 'ENF' AND
1280              (#DATE-11ANS GE #DATE-DEB AND  #DATE-11ANS LE #DATE-FIN)
1290 *
1300 ***** ALORS ON EFFECTUE LES MISES-A-JOURS ... ************************
1310          COMPUTE SCOTYP = +11
1320          UPDATE
1330 ***** ET ON HISTORISE LES VALEURS DANS LE FICHIER HISTORIQUE ... *
1340          PERFORM HISTO
1350          IF ##NUM-ERR NE 0
1360               BACKOUT TRANSACTION
1370          END-IF
1380       END TRANSACTION
1390     END-IF
1400  END-FIND
1410  END-READ
1420 END-SUBROUTINE
1430 *
1440 ********************************************************************
1450 * 3. SI ON DEMANDE MISE-A-JOUR ET LISTE...
1460 *
1470 DEFINE SUBROUTINE MAJ-LISTE
1480 *
1490 READ PERSONNEL-VIEW BY NOM
1500  ACCEPT IF PERSONNEL-VIEW.LSTTYPC < 7 AND PERSONNEL-VIEW.POSTYPC < 31
1510             AND PERSONNEL-VIEW.ENFNBR > 0
1520  FIND PAC-VIEW WITH PACREF = NOPERS
1530   IF PACTYP = 'ENF' AND
1540              (#DATE-11ANS GE #DATE-DEB AND  #DATE-11ANS LE #DATE-FIN)
1550 ********** ON AFFICHE ...
1560          WRITE NOTITLE PERSONNEL-VIEW.NOM PERSONNEL-VIEW.PRENOM
1570                PAC-VIEW.PACPRN PAC-VIEW.PACONA PAC-VIEW.PACREF
1580 **********  ON EFFECTUE LES MISES-A-JOUR... ************************
1590          COMPUTE SCOTYP = +11
1600          UPDATE
1610 **********  ON HISTORISE LES VALEURS MODIFIEES ********************
1620          PERFORM HISTO
1630          IF ##NUM-ERR NE 0
1640               BACKOUT TRANSACTION
1650          END-IF
1660          END TRANSACTION
1670    END-IF
1680  END-FIND
1690 END-READ
1700 END-SUBROUTINE
1710 ********************************************************************
1720 *             POUR L'HISTORISATION                                 *
1730 ********************************************************************
1740 DEFINE SUBROUTINE HISTO
1750 *
1760 *  INITIALISATIONS PARAMETRES HISTORIQUE ...
1770 *
1780 MOVE PERSONNEL-VIEW.IDENT TO       #IDENT
1790 MOVE PERSONNEL-VIEW.MAJEFF TO      #DATE
```

```
1800 MOVE PERSONNEL-VIEW.MAJMODC TO     #MODCOD
1810 MOVE PERSONNEL-VIEW.MAJDEC TO      #DATDEC
1820 MOVE *USER TO                     #USER-OPER
1830 RESET #RESULT #MAJ-SF ##NUM-ERR
1840 MOVE FALSE TO #SW-WARNERR
1850 **** FIN INITIALISATIONS VARIABLES ...
1860 *
1870 **** HISTORISATION DU CODE ALLOC. : SCOTYP.
1880 *
1890 MOVE PAC-VIEW.SCOTYP TO            #FIELDVALUE
1900 MOVE 'SCOTYP' TO                   #FIELDNAME
1910 PERFORM CALL-HISTORY
1920 IF ##NUM-ERR NE 0
1930    ESCAPE ROUTINE
1940 END-IF
1950 *
1960 ****   WARNING COHERENCE HISTORIQUE ...
1970 IF #SW-WARNERR
1980    MOVE 9999 TO ##NUM-ERR
1990 END-IF
2000 END-SUBROUTINE
2010 *
2020 **** SUBROUTINE D'HISTORISATION  CALL-HISTORY
2030 ************************************************
2040 DEFINE SUBROUTINE CALL-HISTORY
2050  CALLNAT 'N##HSTUPF' #FILENAME #IDENT #DATE #FIELDNAME #FIELDVALUE
2060                      #MODCOD #DATDEC #USER-OPER #RESULT #MAJ-SF
2070 *
2080 **** #RESULT ET #MAJ-SF SONT LES RESULTATS FOURNIS PAR CETTE PROCEDURE
2090 ***  ON VA LES TESTER ...
2100 *
2110 IF NOT #RESULT
2120    BACKOUT TRANSACTION             /* PROBLEME SYSTEME...
2130    WRITE 'ERREUR GRAVE DU SYSTEME...CONTACTEZ LE DBA.'
2140    MOVE 123 TO ##NUM-ERR
2150    ESCAPE ROUTINE
2160 END-IF
2170 *
2180 IF NOT #MAJ-SF
2190  IF #FIELDNAME NE 'MAJEFF'
2200    BACKOUT TRANSACTION             /* PROBLEME COHERENCE HISTORIQUE...
2210    WRITE 'ERREUR HISTORIQUE...NON SITUATION FINALE...'
2220    MOVE 123 TO ##NUM-ERR
2230    ESCAPE ROUTINE
2240    END-IF
2250 END-IF
2260 END-SUBROUTINE                     /* CALL-HISTORY
2270 *  ************************************************************
2280 END
***** End of List *****
```

# CONCLUSIONS

Après avoir effectué l'analyse des deux systèmes, un certain nombre de conclusions peuvent être émises.

Au cours de cette étude, nous avons pu constater que même avant la phase d'installation, certains choix organisationnels (à moyen et à long terme) doivent préceder les choix purements techniques. C'était le cas, par exemple, lorsque nous avons soulevé le problème concernant le choix de la "distanciation" à prendre, ou à ne pas prendre, vis-à-vis de l'adaptation d'Arpege au Conseil: plus on changerait sa structure (car cela correspondrait mieux aux besoins du Conseil), plus difficile s'avererait l'intégration des phase futures menées au préalables par le P.E.
Un autre exemple, est le problème organisationnel posé par l'arrivée au Conseil d'une gestion des postes sous Arpege. Alors que cette dernière est gérée actuellement de manière "rudimentaire" au Conseil.
On comprend donc aisement que la phase d'installation doit être précedée d'une phase d'*analyse de l'existant* de manière telle que les capacités du nouveau système **recouvrent (et anticipent)** les besoins informationnels (présents et futurs) du Conseil.

Une fois ces réflexions organisationnelles accomplies, nous pouvons nous focaliser sur la phase de conversion d'un système à l'autre. Pour ce faire deux étapes succéssive: l**a conversion des données et la conversion des fonctions.**

Comme nous avons pu le constater, il existe au niveau des données une grande similitude entre les deux systèmes (la raison en est que les besoins informationnels des deux institutions des Communautés Européennes sont quasi-équivalents).

La seconde étape, quant à elle, en raison de la configuration des fonctions du Conseil, dans un premier temps, nous serons amenés à faire *cohabiter* un certain nombre de ces fonctions (celles des deuxième et troisième couches) avec celles de l'application Arpege. Rien ne nous empechera plus tard de convertir ces fonctions satellites en Natural.

Arpege caracterisé principalement par une architecture modulaire, une flexibilité d'adaptation et une documentation détaillée, nous laisse penser que ce S.I. s'adapterait avec aisance.

L'apport d'Arpege au Conseil est synonyme de nouvelles perspectives. Outre sa convivialité, une gestion des postes dynamique ainsi qu'un traitement de l'historique complet et fiable, ce système offrira aux analystes et programmeurs du Conseil un espace de développement plus "ouvert" au moyen du SGBD Adabas et du langage de programmation Natural.

# BIBLIOGRAPHIE


**[DATE]** *"An introduction to Database systems"*
**C. J. Date**
Addison-Wesley Publishing Company 1983. Volume 1 & 2.

**[DELOBEL&ADIBA]** *"Relational Database systems"*
**C. Delobel & M. Adiba**
Elsevier Science Publishers B.V. 1985

*"The selection of Database software"*
**B. Davis**
NCC Pulications- The National
Computing Center Ltd. 1977

*"Database Techniques. Software selection & System development"*
BIS Applied System Ltd. Q.E.D. Information Science 1980

*"A relational Database management system"*
**A.T.F. Hutt**
John Wiley & Sons 1979

*"New directions for Database systems"*
**G. Ariav & James Clifford**
Ablex Publishing Corporation 1986

*"Database Management"*
**J.W. KLIMBIE & K.L. KOFFEMAN**
North Holland publishing Company 1974

*"Computer Database organisation"*
**James Martin**
Prentice Hall, Inc. 1975

*"Utilisation de Bases de données dans les entreprises"*

**Guy Louis-Gavet**

Editions A.I.D.E. Tome 1. 1981

# ANNEXES

**Annexes 1**: Liste des zones du Signaletic

**Annexes 2**: Nomenclature Arpege

**Annexes 3**: Equations de cohérence pour Arpege

**Annexes 4**: Code source de GLMENS

**Annexes 5:** Listing des résultats donnés par GLMENS